

Appliance Administration Manual

Version 8.13.0 Revision 1.0.0



Table of Contents

Chapter 1 - Introduction	12
About this Manual	12
About the Appliance	12
Latest Version Release Notes	12
Appliance Configuration Overview	13
Appliance Security	14
Security Mode	14
Passwords	14
The "root" Linux account	14
The "loadbalancer" WebUI account	15
Ports Used by the Appliance	15
Additional Information	15
Deployment Guides	15
Quick Start & Configuration Guides	15
Contacting Support	16
Chapter 2 - Load Balancing Concepts	17
Load Balancing - the Basics	17
Supported Protocols	17
Layer 4 & Layer 7	17
What are VIPs and RIPs?	17
What is a Floating IP Address?	17
Load Balancing Algorithms	17
Round Robin / Weighted Round Robin	17
Least Connection / Weighted Least Connection	18
Destination Hashing	18
Real Server Agent	18
Layer 4 vs Layer 7	18
The Basics	18
Performance	18
Persistence	18
Real Server Changes	19
Transparency	19
Other Considerations	19
Does Your Application Cluster Correctly Handle its own State?	19
Replication Solutions for Shared Data	19
Solutions for Session Data	19
What if Your Application is not Stateless?	20
Default Persistence Options	20
What are Your Objectives?	20
Loadbalancer.org Terminology	21
Chapter 3 - Topologies & Load Balancing Methods	23
One-Arm and Two-Arm Topologies	23
Supported Load Balancing Methods	23
Layer 4 DR Mode	25
Layer 4 NAT Mode	26
Layer 4 SNAT Mode	28
Layer 7 SNAT Mode	29
Which Load Balancing Method Should I Use?	30

Mode Summary	30
Our Recommendation	30
Chapter 4 - Appliance Fundamentals	31
Hardware Appliance Installation	31
Virtual Appliance Installation	31
Supported Hypervisors	31
Host Requirements	32
Virtual Hardware Resource Allocations	32
Download & Extract the Appliance	32
Hypervisor Deployment	33
VMware Host Client	33
VMware vSphere Client	33
VMware Workstation Player	34
VMware Tools / Open VM Tools	34
Microsoft Hyper-V	35
Linux Integration Services	35
KVM	35
Nutanix AHV	36
XEN	36
Cloud Appliance Installation	36
Configuring Initial Network Settings	36
Appliance Access & Configuration Methods	41
Accessing the Appliance WebUI	41
Main Menu Options	42
"Root" User Access	43
Keyboard Layout	43
Chapter 5 - Appliance Management	44
Installing the License Key	44
Appliance Software Update	45
Determining the Current Software Version	45
Creating a Backup / Checkpoint	45
Online Update	45
Configuring Online Update	45
Manual Check for Updates	45
Auto-Check for Updates	46
Offline Update	46
Updating a Clustered Pair	47
Network Configuration	48
Physical/Virtual Adapters	48
Configuring IP Addresses	48
Configuring Bonding	49
Configuring VLANs	51
Interface Offloading	52
Configuring MTU Settings	52
Configuring Default Gateway & Static Routes	53
Management Gateway	53
Configuration Example	54
Policy Based Routing (PBR)	54
Configuring Hostname & DNS	55
Service Socket Addresses	56
Appliance Security Features	57

Security Mode	57
Users & Passwords	58
Linux root Account	58
WebUI User Accounts	59
External Authentication	61
Adding Additional Users	64
Firewall Configuration	64
Firewall Lock-down Wizard	65
Conntrack Table Size	67
Appliance Security Lockdown Script	67
SSH Keys	69
System Date & Time Configuration	70
Auto Configuration using NTP Servers	70
Manual Configuration	71
Appliance Internet Access via Proxy	71
SMTP Relay Configuration	72
Syslog Server Configuration	72
SNMP Configuration	73
Running OS Level Commands	75
Portal Management & Appliance Adoption	75
Restarting & Reloading Services	80
Appliance Restart & Shutdown	82
Restoring Manufacturer's Settings	82
Using the WebUI	83
Using the Console / SSH Session	83
Chapter 6 - Configuring Load Balanced Services	84
Introduction	84
Layer 4 Services	84
The Basics	84
Creating Layer 4 Virtual Services	84
Configuring a New Layer 4 VIP	84
Duplicating an Existing Layer 4 VIP	86
Modifying a Layer 4 VIP	86
Creating Layer 4 Real Servers (RIPs)	89
DR Mode Considerations	90
The ARP Problem	90
Detecting the ARP Problem	91
Solving the ARP Problem for Linux	91
Solving the ARP Problem for Solaris	97
Solving the ARP Problem for Mac OS X/BSD	97
Solving the ARP Problem for Windows Servers	98
Solving the ARP Problem - Possible Side Effect for Windows 2012 & Later	104
Other Windows Settings that May Cause Issues	106
Configuring Your Application to Respond to Both the RIP and VIP	108
Windows Firewall Settings	109
NAT Mode Considerations	109
NAT Mode Potential Issues	109
One-Arm (Single Subnet) NAT Mode	110
Firewall Marks	111
Firewall Marks - Auto Configuration	111
Firewall Marks - Manual Configuration	113

Layer 4 - Advanced Configuration	116
Layer 7 Services	117
The Basics	117
Creating Layer 7 Virtual Services (Using the Wizard)	118
Creating Layer 7 Virtual Services	120
Configuring a New Layer 7 VIP	120
Duplicating an Existing Layer 7 VIP	122
Modifying a Layer 7 VIP	122
ACLs (Access Control Lists)	132
Modifying HTTP Header Fields	142
HTTP Header Field Modification Examples	145
Creating Layer 7 Real Servers (RIPs)	146
Layer 7 - Custom Configurations	148
Configuring Manually Defined Virtual Services	148
Transparency at Layer 7	150
Enabling Transparency	151
Inserting Headers	151
Using TProxy to modify the Source IP Address	152
Configuration Examples	153
Layer 7 - Advanced Configuration	157
HAProxy Status Codes	160
Floating IPs	161
SSL Termination	162
Concepts	162
SSL Termination on the Real Servers (SSL Passthrough)	162
SSL Termination on the Load Balancer (SSL Offloading)	163
Using STunnel or Pound to Terminate SSL	163
Using HAProxy to Terminate SSL	164
Certificates	164
Let's Encrypt	168
Creating a SSL Termination	168
Server Name Indication (SNI)	174
SSL Termination on the Load Balancer with Re-encryption (SSL Bridging)	175
Using STunnel or Pound to Terminate SSL	175
Using HAProxy to Terminate SSL	176
Enabling SSL Re-Encryption	176
SSL - Advanced Configuration	177
Pound Global Settings	177
STunnel Global Settings	178
Mutual Transport Layer Security (mTLS)	178
Configuring mTLS	178
Upload the CA Certificate	178
Front-end mTLS	179
Back-end mTLS	180
Certificate Revocation List (CRL)	182
HTTP to HTTPS Redirection	182
When Terminating SSL on the Real Servers	182
When Terminating SSL on the Load Balancer	183
Using STunnel or Pound	183
Using HAProxy	183
Server Feedback Agent	184

Windows Agent	185
Installing the Agent	185
Controlling the Agent	185
Linux/Unix Agent	185
Installation & Testing	185
HTTP Server	186
Configuring VIPs To Use The Agent or HTTP Server	186
Global Server Load Balancing (GSLB)	187
Key Concepts	187
Key features	187
GSLB Configuration	187
GSLB Service IP Address & Port	192
External Health Check Scripts (GSLB)	192
GSLB Advanced Configuration	195
GSLB Multi-site Example	195
Conceptual Overview	195
Appliance Configuration	197
DNS Server Configuration	201
GSLB Diagnostics	201
Configuring the Appliance via CLI, API & Direct Service Calls	203
Command Line Interface (CLI)	203
LBCLI Argument Syntax	203
LBCLI Command Reference	204
Running lbcli from a remote Linux Host	251
Running lbcli from a remote Windows Host	251
Application Programming Interface (API)	251
Enabling the API	251
API Endpoint	252
Creating the JSON Request	252
JSON Syntax Validation	253
Sending API Requests	254
Using ipvsadm to configure Layer 4 Services	255
Using Linux socket commands to configure Layer 7 Services	256
Chapter 7 - Web Application Firewall (WAF)	258
Introduction	258
Implementation Concept	259
Creating a New WAF Gateway	260
Step 1 - Create the Layer 7 VIP	260
Step 2 - Define the Associated Real Servers (RIPs)	260
Step 3 - Define the WAF Gateway	261
Step 4 - Reload Services to Apply the New Settings	261
Step 5 - View Configured Services	261
Fail Open and Fail Closed Designs	262
Configuring a WAF Gateway for Fail Open Operation (Default)	263
Configuring a WAF Gateway for Fail Closed Operation	263
WAF Gateway Settings	264
Disable Web Application Firewall	264
Ruleset	265
Paranoia Level	265
Rule Engine Traffic Blocking	265
Process Request Data	265

Process Response Data	266
Inbound Anomaly Score	266
Outbound Anomaly Score	266
Audit Mode	266
WAF Proxy Timeout	266
Enable Cache Acceleration	267
Double Login Enable	267
WAF - Advanced Configuration	268
PCRE Match Limit	268
PCRE Match Limit Recursion	268
Working With the Core Rule Set	268
What is the Core Rule Set?	269
Core Rule Set Map	269
Anomaly Scoring	269
Overview of Anomaly Scoring	269
How Anomaly Scoring Mode Works	269
Summary of Anomaly Scoring Mode	270
Anomaly Score Thresholds	270
Severity Levels	271
Paranoia Levels	271
Introduction to Paranoia Levels	272
Description of the Four Paranoia Levels	272
Choosing an Appropriate Paranoia Level	273
Setting the Paranoia Level	273
How Paranoia Levels Relate to Anomaly Scoring	274
False Positives and Tuning	274
What are False Positives?	274
Example False Positive	274
Why are False Positives a Problem?	275
Tuning Away False Positives	276
Directly Modifying CRS Rules	276
Rule Exclusions	276
Adding Custom WAF Configuration	282
Per-Transaction Resource Limits	283
Default Limits	283
Example 1	284
Example 2	284
Handling Large HTTP Requests	285
Large File Uploads	285
Requests Containing Large Amounts of Non-File Upload Data	286
Requests with Massive Header Fields	286
WAF Gateway Error Logs	287
Logging Mechanism Overview	287
Viewing the Error Logs	287
Default View	287
Simple View	288
Breakdown View	288
Fixes View	289
Breakdown of a Log Entry	289
Chapter 8 - Real Server Health Monitoring & Control	293
Configuring Health Checks	293

Health Checks for Layer 4 Services	293
Health Checks for Layer 7 Services	296
External Health Check Scripts	299
Default Scripts	300
Adding Additional Health Check Scripts	300
Using Script Templates	300
Uploading External Files	301
Testing External Health Check Scripts at the Command Line	302
Simulating Health Check Failures	303
Disabling Health Checks	303
Fallback Server	303
Local Fallback Server	304
Using a Separate Dedicated Server	304
Using a Layer 7 VIP	305
Configuring A Real Server as the Fallback Server	305
Configuring Primary / Secondary Real Servers	305
Configuring Email Alerts for Virtual Services	305
Layer 4	305
Global Layer 4 Email Settings	305
VIP Level Settings	306
Configuring a Smart Host (SMTP relay)	307
Layer 7	307
Real Server Monitoring & Control using the System Overview	308
Real Server Monitoring	308
Real Server Control	309
Ordering of VIPs	310
Sort by Column	310
Drag & Drop	311
Real Server Monitoring & Control using the HAProxy Statistics Report	311
Real Server Monitoring	311
Real Server Control	312
Chapter 9 - Appliance Clustering for HA	314
Introduction	314
Clustered Pair Concepts	314
Primary/Secondary Operation	314
Pair Communication	314
Heartbeat	314
Primary Secondary Replication	315
Settings that are NOT Replicated to the Secondary Appliance	315
Manually Forcing Appliance Synchronization	316
To Create an HA Pair (Add a Secondary)	316
To Break an HA Pair (Remove a Secondary)	318
Promoting a Secondary to Primary	319
Configuring Heartbeat	320
Communication Method	321
Peer Failure Detection	322
Routing Failure Detection	322
Email Alerts	322
Fail-back Settings	322
Configuring a Smart Host (SMTP relay)	323
Connection State & Persistence Table Replication	323

Layer 4 VIPs	323
Layer 7 VIPs	325
Clustered Pair Diagnostics	325
Heartbeat State Diagnostics	325
Split Brain Scenarios	326
Forcing Primary/Secondary Failover & Failback	327
Testing & Verifying Primary/Secondary Replication & Failover	328
Heartbeat Log	330
Late Heartbeats	331
Chapter 10 - Configuration Examples	332
Introduction	332
Initial Network Settings & WebUI Access	332
Example 1 - One-Arm DR Mode (Single Appliance)	332
Configuration Overview	332
Verify Network Settings	332
Configure the Virtual Service (VIP)	333
Configure the Real Servers (RIPs)	333
Configure the Real Backend Servers for DR Mode	334
Basic Testing & Verification	334
Example 2 - Two-Arm NAT Mode (Clustered Pair)	334
Configuration Overview	335
Configure the Primary's Network Settings	335
Configure the Secondary's Network Settings	336
Configure the Virtual Service (VIP) using the Primary's WebUI	336
Configure the Real Servers (RIPs) using the Primary's WebUI	337
Create the HA Clustered Pair using the Primary's WebUI	338
Configure the Real Backend Servers for NAT Mode	340
Basic Testing & Verification	340
Example 3 - One-Arm SNAT Mode & SSL Termination (Single Appliance)	340
Configuration Overview	341
Verify Network Settings	341
Configure the Virtual Service (VIP)	342
Configure the Real Servers (RIPs)	342
Configure SSL Termination	343
Basic Testing & Verification	344
Example 4 - Load Balancing FTP	344
Layer 4	344
Layer 7	345
Active Mode	345
Passive Mode	347
Chapter 11 - Diagnostics & Troubleshooting	351
The Appliance	351
Resource Utilization	351
Administration Log	351
Apache Logs	351
Apache Access Log	351
Apache Error Log	351
Layer 4 & Layer 7 Virtual Services	351
Checking Service State using the System Overview	351
VIP(s) Fail to appear in the System Overview	352
VIPs & RIPs are Green but Users Can't Connect	353

General	353
Layer 7 VIPs	353
Layer 4 VIPs	353
Diagnosing Real Server Issues	354
Requests are Not being Load Balanced as Expected	355
Log File - Layer 4 Virtual Services	356
Interpreting the log file	356
Log File - layer 7 Virtual Services	356
Interpreting the log file	357
SSL Termination (Pound)	358
SSL Termination (STunnel)	358
WAF	359
Heartbeat	359
Shuttle	359
Chapter 12 - Monitoring & Reporting	360
SNMP Reporting	360
MIB Files	360
SNMP for Layer 4 Services	360
Monitoring Layer 4 VIPs & RIPs using SNMP	361
SNMP for Layer 7 Services	362
Monitoring Layer 7 VIPs & RIPs using SNMP	362
SNMPv3	364
Reports	364
Layer 4 Status	364
Layer 4 Traffic Rate	365
Layer 4 traffic Counters	366
Layer 4 Current Connections	367
Layer 4 Current Connections (Resolve Hostnames)	368
Layer 7 Status	368
Layer 7 Stick Table	368
GSLB Reports	369
Graphs	369
Resource Utilization Graphs	369
Virtual Service & Real Server Graphs	371
Graphing Options	372
Layer 7 (HAProxy) Prometheus Exporter	373
Grafana	374
Chapter 13 - Useful Tools & Utilities	375
Useful Diagnostics Tools	375
Netstat	375
Telnet	375
Tcpdump	376
Ethtool	376
Nmap	377
Iptraf	377
Wireshark	378
Windows Specific Tools	378
Microsoft Network Monitor	378
WinSCP	378
PuTTY	378
Chapter 14 - Backup & Restore and Disaster Recovery	379

Backup & Restore	379
Backup	379
Restore	379
Restore System Defaults	380
Checkpoints	381
Template Deployment	382
Disaster Recovery	383
Being Prepared - Creating Backups	383
Firmware Recovery using a USB Memory Stick	383
Disaster Recovery After Node (Primary or Secondary) Failure	386
Chapter 15 - Technical Support	390
Introduction	390
WebUI Support Options	390
Contact Us	390
Technical Support Download	390
Useful Links	391
Remote Support	391
Live Chat	391
Appendix	393
Front & Rear Panel Layouts	393
Enterprise Prime	393
Enterprise Flex	393
Enterprise Max	393
IPMI Configuration	393
IPMI Network Settings	393
Using the Setup Utility	394
Using the IPMI Web Interface	395
Technical Support / More Information	395
iDRAC Configuration	395
iDRAC Network Settings	395
Using System Setup	396
Using the iDRAC Web Interface	398
Technical Support / More Information	398
Appliance IPv4 Address Format (CIDR notation)	398
Appliance Configuration Files & Locations	399

Chapter 1 - Introduction

About this Manual

This document covers all required administration information for v8.13.x Loadbalancer.org appliances.

About the Appliance

The core software is based on LBOS-7 which is a customized Linux build maintained by Loadbalancer.org, LVS, Ldirectord, Linux-HA, HAProxy & STunnel. Full root access is provided which enables complete control of all settings.

The appliance is available in the following formats: hardware, virtual (VMware, HyperV, KVM, Nutanix & XEN) and cloud based (Amazon, Azure & GCP).

The appliance can be deployed as a single unit or as a clustered pair.

Note

Loadbalancer.org always recommend that clustered pairs should be used where possible for high availability and resilience, this avoids introducing a single point of failure to your network. For more information on configuring an HA pair, please refer to [Chapter 9 - Appliance Clustering for HA](#).

Latest Version Release Notes

The latest version of the appliance (v8.13.0) includes the following new features & improvements, bug fixes and security updates:

New Features & Improvements

- **HTTP/2 Client Certificates** : Added option to allow connections to be made by clients presenting invalid certificates so that ACL rules may be used to handle such connections.
- **New Port Range Syntax** : Added the ability to specify port ranges with holes using the new ! difference operator e.g.: 50-500!(80,443) (ports 50 to 500 except 80 and 443).
- **Multi-port SNAT** : Layer 4 SNAT services may now be given port ranges.
- **Improved SNAT performance** : Layer 4 SNAT rules are now created in chains per-Virtual Service, improving the speed of their creation, deletion and execution.
- **Hysteresis for communications with peer** : Failures to communicate with the peer node are now remembered between calls and no further checks are made until a timer expires eliminating costly checks that slow down operations when one node has failed.
- **Improved Layer 4 Health Checks** : It is now possible to check the result of a non-200 response to an HTTP health check at layer 4 without using an external health check script.

Bug Fixes

- **Ldirectord** : Virtual Services with a check type of "No checks, always On" will no longer have a fallback server erroneously added to the pool of available servers when the service restarts or reloads.



- **Certificate Usage** : When displaying the certificate usage, if the node is in a pair the peer is consulted to obtain a list of certificates used by its system services.
- **WAFs and duplicated services** : Made it so that it is no longer possible to associate WAF Gateways with unedited, duplicated Virtual Services.
- **UI Improvement** : The check port field is no longer displayed when an external health check is selected for layer 4 Virtual Services.
- **Real Server weights** : Halting or draining a Real Server multiple times no longer overwrites the previous weight.
- **UI Improvement** : The duplicated cookie name warning is no longer triggered by comparing a Virtual Service with itself.
- **UI Improvement** : The page title now shows the hostname, IP address and version of the appliance.
- **UI Improvement** : Unedited, duplicate Virtual Services no longer cause an indexing issue on the overview page.
- **PBR Rules** : Made it so that full-matches are made against the name of the PBR rule when deleting PBR rules.
- **Let's Encrypt Script** : Made it so that certificates obtained via the lb-letsencrypt.sh script are installed using LBCLI removing the need to provide credentials for the WebUI.
- **Layer 7 Persistence Cookies** : Fixed a regression where persistence cookies that were not marked with the secure attribute were sent with the SameSite=None attribute causing browsers to reject them.
- **GSLB via WARP** : Fixed an issue preventing GSLB objects from being edited via WARP.

Security Updates

- Updated OpenSSH to Version 9.9p2 addressing CVE-2025-26465 and CVE-2025-26466.

Appliance Configuration Overview

Initial network configuration is carried out at the console using the [Network Setup Wizard](#). Once the wizard has been run, load balanced services can be configured using the WebUI - either using the [Setup Wizard](#) which can be used to configure layer 7 Virtual Services (VIPs) and the associated Real Servers (RIPs) or by manually configuring the required services.

By default, the WebUI is accessible on HTTPS port **9443**, this can be changed if required. For more information, please refer to the "Appliance Security" section below.

We always recommend that where possible two appliances are deployed as a clustered pair for high availability and resilience, this avoids introducing a single point of failure to your network.

We recommend that the Primary appliance is fully configured first, then the Secondary appliance can be added to create an HA pair. Once the HA pair is configured, load balanced services must be configured and modified on the Primary appliance. The Secondary appliance will be automatically kept in sync. For more information on configuring an HA pair, please refer to [Chapter 9 - Appliance Clustering for HA](#).



Note

For Enterprise Azure, the HA pair should be configured first. In Azure, when creating a VIP using



an HA pair, two private IPs must be specified – the first for the VIP when it's active on the Primary and the other for the VIP when it's active on the Secondary. Configuring the HA pair first, enables both IPs to be specified when the VIP is created.

Appliance Security

Note

For full details of each security mode and all other security related features, please refer to [Appliance Security Features](#).

Security Mode

To control how the appliance is accessed and which features are enabled, 3 security modes are provided:

- **Custom** - In this mode the security options can be configured to suit your requirements
- **Secure - (Default)** - In this mode:
 - "root" user console access & SSH password access are disabled
 - WebUI connections are forced to use HTTPS
 - Access to the *Local Configuration > Execute shell command* menu option is disabled
 - The Firewall Script & the Firewall Lockdown Wizard Script cannot be edited
- **Secure - Permanent** - This mode is the same as **Secure** but once set it cannot be changed

(!) Important Setting the security mode to **Secure - Permanent** is irreversible.

To configure the Security Mode:

1. Using the WebUI, navigate to: *Local Configuration > Security*.
2. Select the required *Appliance Security Mode*.
3. Click **Update**.

Passwords

The password for the "root" user Linux account and the "loadbalancer" WebUI user account are set during the Network Setup Wizard.

Note

The passwords for the cloud products are either set to a default value or are configured during instance deployment. Also, for Enterprise AWS and Enterprise Azure it's not possible to directly log in as "root". For more details, please refer to the relevant [Quick Start Configuration Guide](#).

The "root" Linux account

As explained in [Security Mode](#) above, "root" user console & SSH password access are disabled by default. Once enabled, the "root" user password can be changed at the console or via an SSH session using the following command:



```
# passwd
```

The "loadbalancer" WebUI account

This can be changed using the WebUI menu option: **Maintenance > Passwords**.

Ports Used by the Appliance

By default, the appliance uses the following TCP & UDP ports:

Protocol	Port	Purpose
TCP	22 *	SSH
TCP & UDP	53 *	DNS / GSLB
TCP & UDP	123	NTP
TCP & UDP	161 *	SNMP
UDP	6694	Heartbeat between Primary & Secondary appliances in HA mode
TCP	7778	HAProxy persistence table replication
TCP	9000 *	Gateway service (Centralized/Portal Management)
TCP	9080 *	WebUI - HTTP (disabled by default)
TCP	9081 *	Nginx fallback page
TCP	9443 *	WebUI - HTTPS
TCP	25565 *	Shuttle service (Centralized/Portal Management)

Note

The ports used for SSH, GSLB, SNMP, the WebUI, the fallback page, the gateway service and the shuttle service can be changed if required. For more information, please refer to [Service Socket Addresses](#).

Additional Information

Deployment Guides

Comprehensive deployment guides are available that focus on load balancing specific applications. They cover the configuration of the load balancer and also any application specific configuration changes that are required to enable load balancing. All guides are available on our website at the following URL:

<https://www.loadbalancer.org/support/deployment-guides/>.

Quick Start & Configuration Guides

The following related documentation may also be useful:

- [Quick Start Guide - Hardware & Virtual](#)
- [Quick Start Configuration Guide - Amazon AWS](#)



- [Quick Start Configuration Guide - Microsoft Azure](#)
- [Quick Start Configuration Guide - Google Cloud Platform](#)

Contacting Support

If you have any questions regarding the appliance or need assistance with load balancing your application, please don't hesitate to contact support@loadbalancer.org.



Chapter 2 - Load Balancing Concepts

Load Balancing - the Basics

Loadbalancer.org appliances enable two or more servers to be combined into a cluster. This enables inbound requests to be distributed across multiple servers which provides improved performance, reliability and resilience. Appliances can also be deployed as a clustered pair (our recommended solution) which creates a highly-available configuration.

Supported Protocols

Loadbalancer.org appliances are able to load balancer virtually any TCP or UDP based protocol including HTTP, HTTPS, FTP, SMTP, RDP, SIP, IMAP, POP, DNS etc. etc.

Layer 4 & Layer 7

Load balancing at layer 4 and layer 7 is supported. LVS (Linux Virtual Server) is utilized at layer 4 whilst HAProxy is used at layer 7.

What are VIPs and RIPs?

Load balancer vendors typically use the term Virtual IP address (VIP) to describe the address that the load balanced cluster is accessed from. It's important to understand that "VIP" refers to both the physical IP address and also to the logical load balancer configuration. Likewise, "RIP" refers to both the Real Server's physical IP address and its representation in the logical load balancer configuration.

Note

It's not possible to configure a VIP on the same IP address as any of the network interfaces. This ensures that services are able to "float" (move) between Primary and Secondary appliances when using an HA clustered pair.

What is a Floating IP Address?

A floating IP address is automatically created whenever a VIP is configured. The floating IP address is the same as the VIP address. Since the floating IP must be able to move between the Primary and Secondary appliance, it's not possible to configure a VIP on the same IP address as an interface as mentioned in the note above. Floating IPs can also be manually configured to provide a "floating default gateway" for configurations that require the load balancer to be the default gateway such as layer 4 NAT mode. This allows the default gateway for the Real Servers to be brought up on the Secondary appliance should the Primary fail.

Load Balancing Algorithms

The Loadbalancer.org appliance supports several different load balancing algorithms. Each has its own advantages and disadvantages and it depends on the specific application which is the most appropriate to use. Usually the default method **Weighted Least Connection** is a good solution which works well in most situations. The following sections summarize each method supported.

Round Robin / Weighted Round Robin

With this method, incoming requests are distributed to Real Servers in a sequential manner relative to each Real Server's weight. Servers with a higher weight receive more requests. A server with a weight of 200 will receive 4



times the number of requests than a server with a weight of 50. Weightings are relative, so it makes no difference if Real Server #1 and #2 have weightings of 50 and 100 respectively or 5 and 10.

Least Connection / Weighted Least Connection

With this method, incoming requests are distributed to Real Servers with the fewest connections relative to each Real Server's weight. Servers with a higher weight receive more requests. A server with a weight of 200 will receive 4 times the number of requests than a server with a weight of 50. Again, weightings are relative, so it makes no difference if Real Server #1 and #2 have weightings of 50 and 100 respectively or 5 and 10. This is the default method for new VIPs.

Destination Hashing

With the method, requests are distributed to Real Servers by looking up the destination IP in a static hash table. This algorithm is designed for use with web proxies and is supported with layer 4 DR mode Virtual Services only. For more information on this method, please refer to [Modifying a Layer 4 VIP](#).

Real Server Agent

To compliment the methods above, Loadbalancer.org appliances also support Real Server (i.e backend server) agents. This permits the load balancing algorithm to be dynamically modified based on each Real Server's running characteristics. For example, one Real Server could have a run-away process that is consuming excessive CPU resources or RAM. Without the agent, the load balancer has no way of knowing this and would continue to send requests to the overloaded server based on the algorithm selected. With the agent installed on the Real Server, feedback is provided to the load balancer and the algorithm is then adjusted to reduce requests that are sent to that server. For more information, please refer to [Server Feedback Agent](#).

Layer 4 vs Layer 7

A fundamental choice when setting up the load balancer is whether to configure the services at layer 4 or layer 7.

The Basics

At layer 4 the primary protocols used are TCP and UDP. These protocols are not aware of upper level protocols such as FTP, HTTP, HTTPS, DNS, RDP etc. Therefore the load balancer can only make load balancing decisions based on details available at layers 4 and below such as port numbers and IP addresses. At layer 7, the load balancer has more information to make load balancing related decisions since more information about upper level protocols is available.

Layer 7 load balancing uses a proxy at the application layer (HAProxy). Requests are terminated on the load balancer, and the proxy generates a new request which is passed to the chosen Real Server.

Performance

Due to the increased amount of information at layer 7 and the fact that a proxy is in use, performance is not as fast as at layer 4. If raw throughput is a primary concern, then layer 4 is probably the better choice.

Persistence

Persistence (aka affinity or sticky connections) is the ability to ensure that a specific client connects back to the same server within a specific time limit. It is normally required when the session state is stored locally on the Real



Server rather than in a separate database. At layer 4, Source IP persistence is the only option. At layer 7, additional methods are available such as HTTP cookie persistence where the load balancer sets a cookie to identify the session and Microsoft Connection Broker where the load balancer is able to utilize the redirection token for reconnecting users to existing sessions.

Real Server Changes

For layer 4 DR mode, the "ARP Problem" must be solved - for more information, please refer to [DR Mode Considerations](#) . For layer 4 NAT mode, the default gateway on each Real Server must be the load balancer. For layer 4 SNAT mode and layer 7 SNAT mode the Real Servers do not need to be changed in any way.

Transparency

Transparency refers to the ability to see the originating IP address of the client. For layer 4 DR mode and NAT mode, connections are transparent. For layer 4 SNAT mode and layer 7 SNAT mode, the IP address of the load balancer is recorded as the source address (for Layer 7 SNAT mode, this can also be set to a user configured address). For layer 7 SNAT mode, additional configuration steps can be taken to force the client IP to be logged. Options include using TProxy to re-write the source address or by enabling support for X-Forwarded-For or Proxy Protocol headers. For more information, please refer to [Transparency at Layer 7](#).

Other Considerations

Does Your Application Cluster Correctly Handle its own State?

Load balancers work most effectively if the application servers are completely stateless. This means that if an application server (i.e. Real Server) fails and is automatically taken out of the cluster, then all current user sessions will be transferred to other servers in the cluster without users needing to re-authenticate to the application.

Web based applications are inherently stateless and are an ideal candidate for load balancing. However, do your web servers store files and other information on local drives?

- Images (jpeg, png, gif etc.)
- Files (html, php, asp etc.)

If so, these files need to be on shared storage or they need to be replicated to all nodes in the cluster.

Replication Solutions for Shared Data

On Linux systems you can use the Rsync command to replicate files. On Windows systems Rsync can also be used, although the Robocopy command may be preferred. Typically, you'll upload content to one primary server and then replicate this to the other servers in the cluster.

Solutions for Session Data

Standard ASP and PHP session data is stored locally by default, leaving your session data in a local store will prevent you from implementing seamless application server fail-over in your cluster. If an application server fails, all of the local session data will be lost and your users will need to re-authenticate and possibly lose shopping baskets etc.

This problem can be resolved by implementing a shared persistent data store for the cluster. This is usually



achieved using a shared backend database.

What if Your Application is not Stateless?

Some protocols require state to be maintained such as:

- SSH
- FTP
- SMTP

You may also find that you are unable to modify your HTTP/HTTPS based application to handle shared session data.

For these cases you can use persistence. You lose the ability to have transparent fail-over, but you benefit from increased capacity and manageability.

Default Persistence Options

For layer 4, source IP address persistence can be used and is enabled by default for all new Virtual Services.

For layer 7, the following persistence options are supported:

- Source IP Address
- HTTP Cookie (the default for new Virtual Services)
- Application Cookie
- SSL Session ID
- MS Session Broker
- RDP Client Cookie
- HTTP Cookie with Fallback to Source IP
- X-Forwarded-For with Fallback to Source IP
- Stick On Fallback
- Last Successful

Note

SSL session ID based persistence is useful in certain circumstances, although due to the way some browsers operate - notably older versions of Internet Explorer, the session ID can be renegotiated frequently (every few seconds) which breaks the persistence.

Note

For details of all standard layer 7 persistence options, please refer to [Modifying a Layer 7 VIP](#). It's also possible to configure other custom persistence types if required using the custom configuration option available for layer 7 Virtual Services. For more information, please refer to [Layer 7 - Custom Configurations](#).

What are Your Objectives?



It's important to have a clear focus on your objectives and the required outcome for the successful implementation of your load balancing solution. Are you looking for increased performance, reliability, ease of maintenance or all three?

Performance	A load balancer can increase performance by allowing you to utilize several commodity servers to handle the workload of one application.
Availability	Running an application on one server gives you a single point of failure. Utilizing a load balancer to present multiple servers improves application availability but moves the point of failure to the load balancer. We always advise that you deploy load balancers as clustered pairs to remove this single point of failure. For more information, please refer to Chapter 9 - Appliance Clustering for HA).
Maintenance	Using the appliance, you can easily bring servers on and off line to perform maintenance tasks, without disrupting users.

Note

In order to achieve all three objectives, your application must handle persistence correctly. For more information, please refer to [Does Your Application Cluster Correctly Handle its own State?](#).

Loadbalancer.org Terminology

Load Balancer	An IP based traffic manager for server clusters.
Primary	The normally active appliance in a HA clustered pair of load balancers.
Secondary	The normally passive appliance in a HA clustered pair of load balancers.
Virtual Service	The main building block used to configure load balanced services. It defines the IP address clients connect to, which Real Servers are load balanced and other settings such as health check options, persistence options and timeout settings.
Real Server	The actual backend server being load balanced. Multiple Real Servers are associated with a Virtual Service.
VIP	Virtual IP address - the IP address of the load balanced cluster of RIPv, the address presented to connecting clients. Also refers to the logical load balancer configuration and is used as an acronym for Virtual Service.
RIP	The Real IP address - the IP address of a backend server in the cluster. Also refers to the logical load balancer configuration and is used as an acronym for Real Server.
Floating IP	The Floating IP Address is automatically created whenever a Virtual Service is configured, the floating IP address is the same as the VIP address. It enables services to be moved between the Primary and Secondary appliance.
WebUI / WUI	Web User Interface. Used to configure and manage the appliance.
Layer 4	Part of the seven layer OSI model. Also a descriptive term for load balancing methods that routes packets based on TCP/IP header information.
Layer 7	Part of the seven layer OSI model. Also a descriptive term for a proxy based load balancing method that distributes packets based on the entire TCP/IP header and also the payload information at the application layer.

DR Mode	Direct Routing (aka DSR/Direct Server Return) is a standard layer 4 load balancing technique that distributes packets by altering only the destination MAC address of the packet.
NAT Mode	Network Address Translation is a standard layer 4 load balancing technique that changes the destination of packets to and from the VIP (external subnet to internal cluster subnet).
Layer 4 SNAT Mode	Source Network Address Translation - similar to NAT mode but also modifies the source address of all outgoing traffic to be the load balancer.
Layer 7 SNAT Mode	Source Network Address Translation - the load balancer acts as a proxy for all incoming & outgoing traffic.
SSL Termination	The SSL certificate is installed on the load balancer in order to decrypt HTTPS traffic on behalf of the cluster.
MASQUERADE	Descriptive term for standard firewall technique where internal servers are represented as an external public IP address. Sometimes referred to as a combination of SNAT & DNAT rules.
One-Arm	The load balancer has one physical network card connected to one subnet.
Two-Arm	The load balancer has two interfaces connected to two subnets - this can be achieved by using two network adapters, or by creating VLANs on a single adapter.
GW	The Default Gateway for a backend server in the cluster.
Eth0	The first Ethernet interface. Also known as Gb0 on the Enterprise Flex and Max. Usually used as the internal interface in a two-arm deployment, although this is optional.
Eth0	The second Ethernet interface. Also known as Gb1 on the Enterprise Flex and Max. Usually used as the external interface in a two-arm deployment, although this is optional.
Eth2	The third Ethernet interface.
Eth3	The fourth Ethernet interface.
Eth4	The Fifth Ethernet interface (Enterprise Max only, also depends on choice of adapter cards).
Eth5	The Sixth Ethernet interface (Enterprise Max only, also depends on choice of adapter cards).

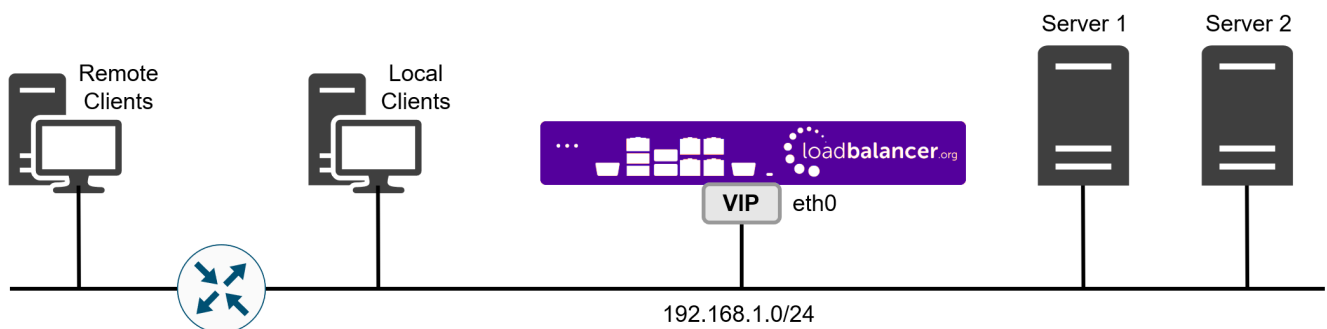
Chapter 3 - Topologies & Load Balancing Methods

One-Arm and Two-Arm Topologies

The number of "arms" is a descriptive term for how many interfaces are used to connect a device to a network. It's common for a load balancer that uses a routing method (NAT) to have a two-arm configuration although one-arm is also supported. Proxy based load balancers (SNAT) commonly use a one-arm configuration although two-arm is also supported.

One-Arm

The VIP and the load balanced servers are located in a single subnet. The load balancer requires a single network interface adapter - **eth0** in the diagram below.

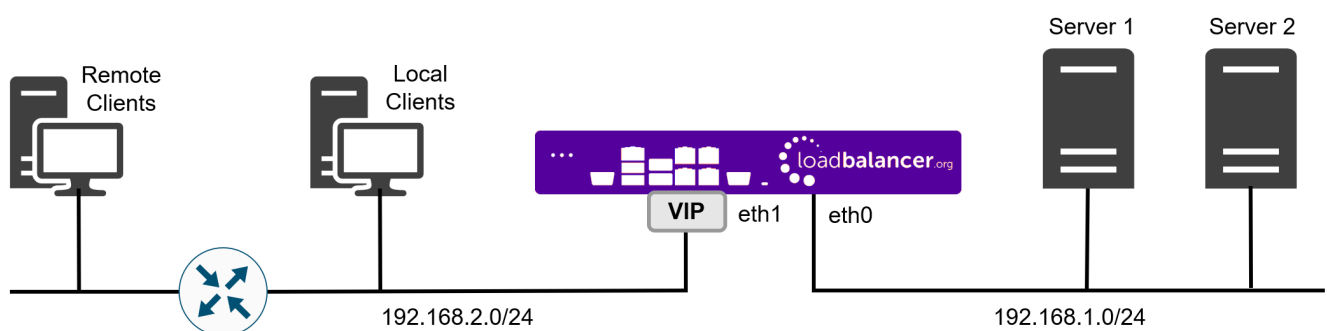


Two-Arm

Here, 2 subnets are used. The VIP is located in one subnet and the load balanced Real Servers are located in the other. The load balancer requires 2 interfaces, one in each subnet as shown in the diagram below.

Note

This can be achieved by using two network adapters, or by creating VLANs on a single adapter.



Note

Typically **eth0** is used as the internal interface and **eth1** is used as the external interface. This is not a requirement - each interface can be used for any purpose.

Supported Load Balancing Methods

The Loadbalancer.org appliance supports multiple load balancing methods/modes. These can be used at the same time or in combination with each other and are described in the table below.



Layer	Method/Mode	Comments	Topology	Note
Layer 4	DR Mode (Direct Routing)	Ultra-fast layer 4 load balancing <ul style="list-style-type: none"> Requires the "ARP problem" to be solved on each Real Server - for more information, please refer to DR Mode Considerations 	One-Arm (*)	1
Layer 4	NAT Mode (Network Address Translation)	Fast Layer 4 load balancing <ul style="list-style-type: none"> The appliance must be the default gateway for the Real Servers 	One-Arm or Two-Arm	1
Layer 4	TUN Mode	Similar to DR mode but works across IP encapsulated tunnels	One-Arm	2
Layer 4	SNAT Mode (Source Network Address Translation)	Fast layer 4 load balancing <ul style="list-style-type: none"> Supports both TCP & UDP Very simple to implement Requires no Real Server configuration changes 	One-Arm or Two-Arm	3
Layer 7	SNAT Mode (Source Network Address Translation using HAProxy)	Allows greater flexibility including full SNAT and remote server load balancing, also supports advanced functionality such as multiple persistence methods, header manipulation and URL rewriting <ul style="list-style-type: none"> Very simple to implement Requires no Real Server configuration changes Not as fast as Layer 4 methods 	One-Arm or Two-Arm	4
Layer 7	SSL Termination (STunnel, Pound & HAProxy)	Typically required to allow cookie persistence, header manipulation and URL rewriting in HTTPS streams <ul style="list-style-type: none"> SSL termination is processor intensive 	One-Arm or Two-Arm	5

(*) DR mode can also be used in a multi-homed configuration where Real Servers are located in different subnets. In this case, the load balancer must have an interface in the same subnet to enable layer 2 connectivity which is required for DR mode to operate.

Notes

1. Recommended for high performance fully transparent and scalable solutions.
2. Only required for DR mode implementations across routed networks (rarely used).
3. Useful when you want to load balance both TCP and UDP but you're unable to use DR mode or NAT mode due to network topology or Real Server related reasons.
4. Used for multiple Microsoft applications such as Exchange, SharePoint, ADFS and RDS.



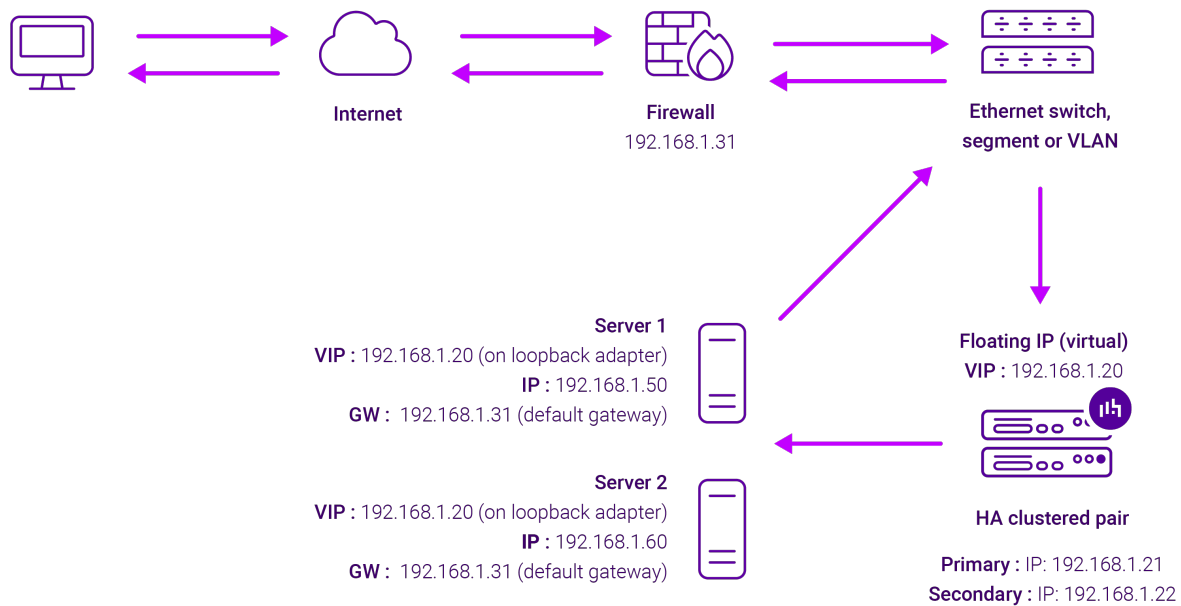
- Where possible, for a scalable solution, terminating SSL on the Real Servers is recommended.

Layer 4 DR Mode

Layer 4 DR (Direct Routing) mode is a very high performance solution that requires little change to your existing infrastructure. The image below shows an example network diagram for this mode.

Note

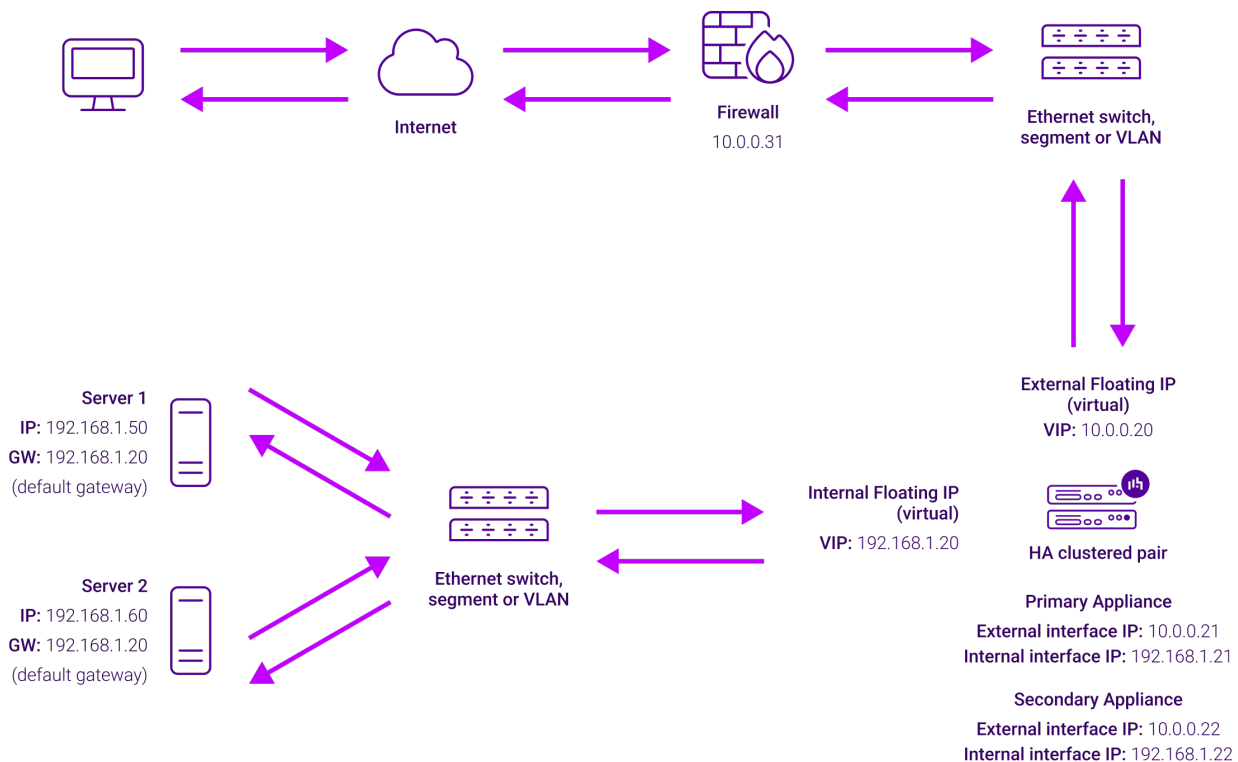
Kemp, Brocade, Barracuda & A10 Networks call this *Direct Server Return* and F5 call it *nPath*.



- DR mode works by changing the destination MAC address of the incoming packet to match the selected Real Server on the fly which is very fast.
- When the packet reaches the Real Server it expects the Real Server to own the Virtual Services IP address (VIP). This means that each Real Server (and the load balanced application) must respond to both the Real Server's own IP address and the VIP.
- The Real Server should not respond to ARP requests for the VIP. Only the load balancer should do this. Configuring the Real Server in this way is referred to as "Solving the ARP Problem". For more information please refer to [DR Mode Considerations](#).
- On average, DR mode is 8 times quicker than NAT mode for HTTP and much faster for other applications such as Remote Desktop Services, streaming media and FTP.
- The load balancer must have an interface in the same subnet as the Real Servers to ensure layer 2 connectivity which is required for DR mode to operate.
- The VIP can be brought up on the same subnet as the Real Servers or on a different subnet provided that the load balancer has an interface in that subnet.
- Port translation is not possible with DR mode, e.g. `VIP:80 → RIP:8080` is not supported.
- DR mode is transparent, i.e. the Real Server will see the source IP address of the client.

Layer 4 NAT Mode

Layer 4 NAT mode is a high performance solution, although not as fast as layer 4 DR mode. This is because real server responses must flow back to the client via the load balancer rather than directly as with DR mode. The image below shows an example network diagram for this mode.



- The load balancer translates all requests from the Virtual Service to the Real Servers.
- NAT mode can be deployed in the following ways:
 - **Two-arm (using 2 Interfaces)** (as shown above) - Here, 2 subnets are used. The VIP is located in one subnet and the load balanced Real Servers are located in the other. The load balancer requires 2 interfaces, one in each subnet.

Note

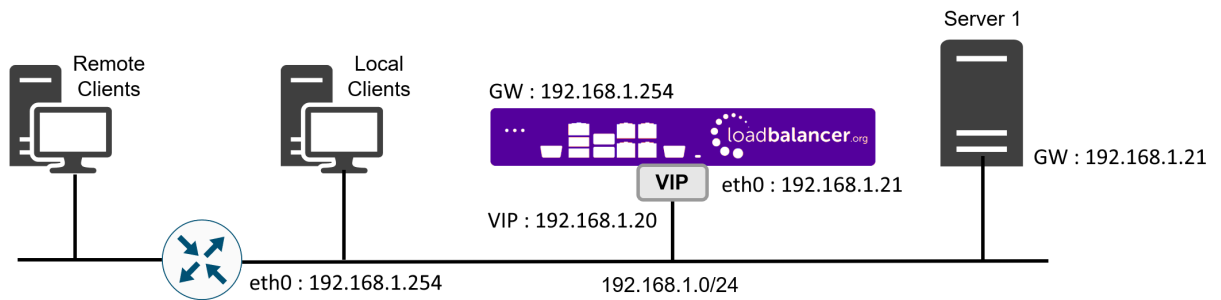
This can be achieved by using two network adapters, or by creating VLANs on a single adapter.

- Normally **eth0** is used for the internal network and **eth1** is used for the external network, although this is not mandatory since any interface can be used for any purpose.
- If the Real Servers require Internet access, **Auto-NAT** should be enabled using the WebUI menu option: **Cluster Configuration > Layer 4 - Advanced Configuration**, the external interface should be selected.
- The default gateway on the Real Servers must be set to be an IP address on the load balancer.

Note

For an HA clustered pair, a floating IP should be added to the load balancer and used as the Real Server's default gateway. This ensures that the IP address can "float" (move) between Primary and Secondary appliances.

- Clients can be located in the same subnet as the VIP or any remote subnet provided they can route to the VIP.
- **One-arm (using 1 Interface)** - Here, the VIP is brought up in the same subnet as the Real Servers.



- To support remote clients, the default gateway on the Real Servers must be an IP address on the load balancer and routing on the load balancer must be configured so that return traffic is routed back via the router.

Note

For an HA clustered pair, a floating IP should be added to the load balancer and used as the Real Server's default gateway. This ensures that the IP address can "float" (move) between Primary and Secondary appliances.

- To support local clients, return traffic would normally be sent directly to the client bypassing the load balancer which would break NAT mode. To address this, the routing table on the Real Servers must be modified to force return traffic to go via the load balancer. For more information please refer to [One-Arm \(Single Subnet\) NAT Mode](#).
- If you want Real Servers to be accessible on their own IP address for non-load balanced services, e.g. RDP, you will need to setup individual SNAT and DNAT firewall script rules for each Real Server or add additional VIPs for this.
- Port translation is possible with Layer 4 NAT mode, e.g. VIP:80 → RIP:8080 is supported.
- NAT mode is transparent, i.e. the Real Server will see the source IP address of the client.

NAT Mode Packet re-Writing

In NAT mode, the inbound destination IP address is changed by the load balancer from the Virtual Service IP address (VIP) to the Real Server. For outbound replies the load balancer changes the source IP address of the Real Server to be the Virtual Services IP address.

The following table shows an example NAT mode setup:

Protocol	VIP	Port	RIP	Port
TCP	10.0.0.20	80	192.168.1.50	80

In this simple example all traffic destined for IP address 10.0.0.20 on port 80 is load-balanced to the real IP address 192.168.1.50 on port 80.



Packet rewriting works as follows:

1) The incoming packet for the web server has source and destination addresses as:

Source	x.x.x.x:34567	Destination	10.0.0.20:80
--------	---------------	-------------	--------------

2) The packet is rewritten and forwarded to the backend server as:

Source	x.x.x.x:34567	Destination	192.168.1.50:80
--------	---------------	-------------	-----------------

3) Replies return to the load balancer as:

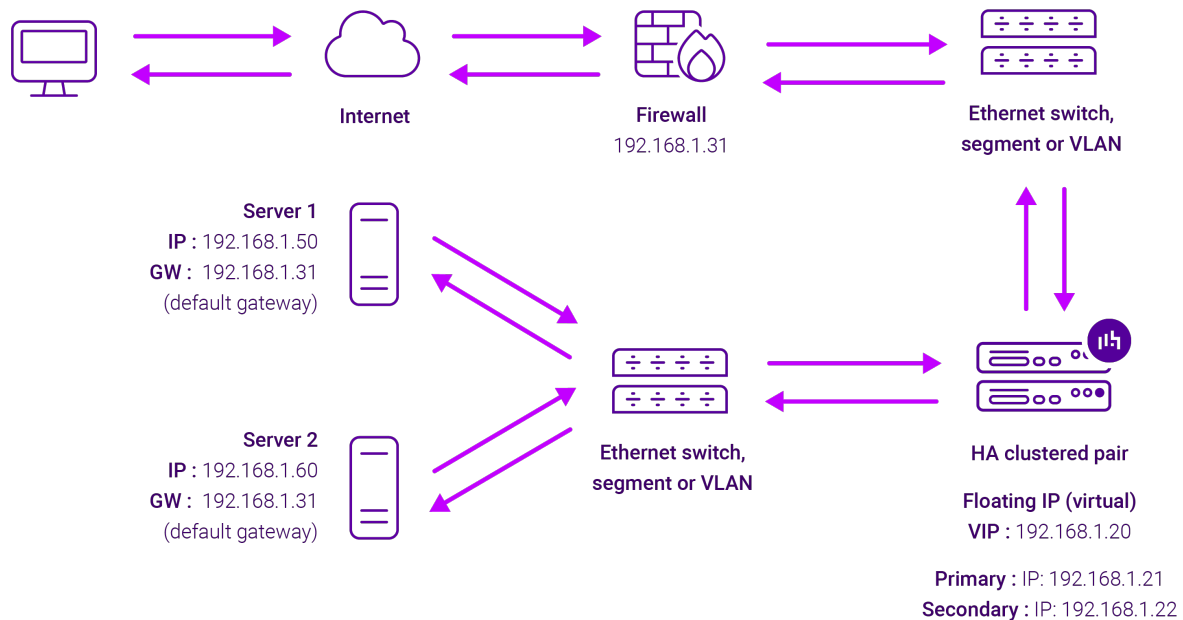
Source	192.168.1.50:80	Destination	x.x.x.x:34567
--------	-----------------	-------------	---------------

4) The packet is written back to the VIP address and returned to the client as:

Source	10.0.0.20:80	Destination	x.x.x.x:34567
--------	--------------	-------------	---------------

Layer 4 SNAT Mode

Layer 4 SNAT mode is a high performance solution, although not as fast as Layer 4 NAT mode or Layer 4 DR mode. The image below shows an example network diagram for this mode.



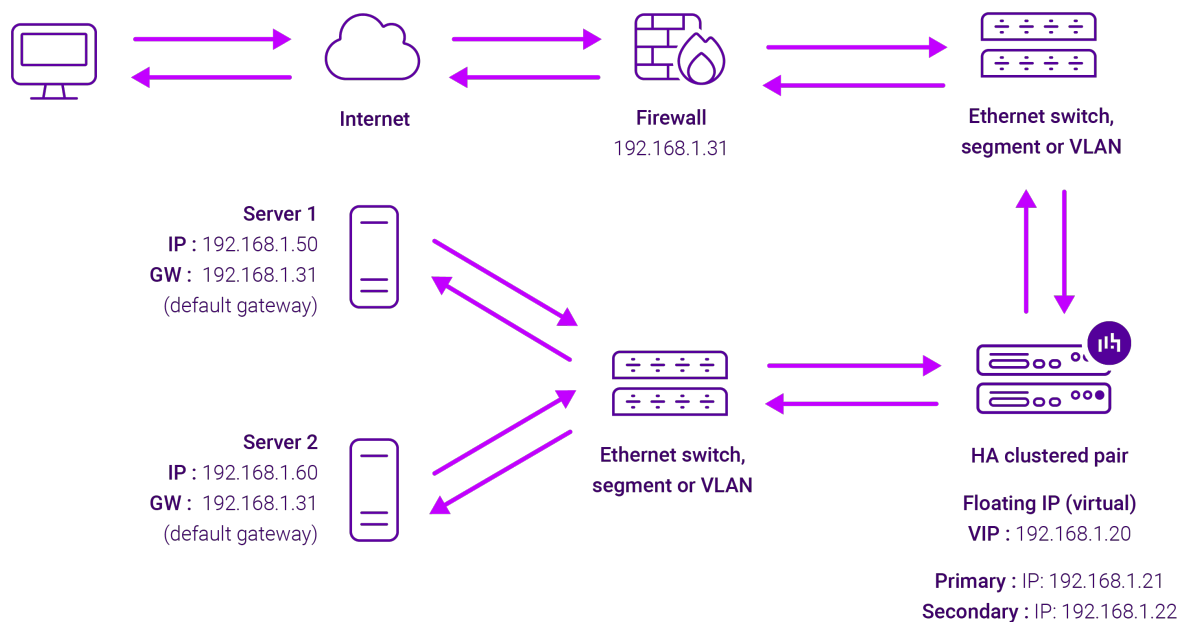
- Real Servers in the cluster can be on any accessible network including across the Internet or WAN.
- Layer 4 SNAT mode is not transparent, an iptables SNAT rule translates the source IP address to be the load balancer rather than the original client IP address.
- Layer 4 SNAT mode can be deployed using either a one-arm or two-arm configuration. For two-arm deployments, **eth1** is typically used for client side connections and **eth0** is used for Real Server connections,

although this is not mandatory since any interface can be used for any purpose.

- Requires no mode-specific configuration changes to the load balanced Real Servers.
- Port translation is possible with Layer 4 SNAT mode, e.g. VIP:80 → RIP:8080 is supported.
- You should not use the same RIP:PORT combination for layer 4 SNAT mode VIPs and layer 7 SNAT mode VIPs because the required firewall rules conflict.

Layer 7 SNAT Mode

Layer 7 SNAT mode uses a proxy (HAProxy) at the application layer. Inbound requests are terminated on the load balancer and HAProxy generates a new corresponding request to the chosen Real Server. As a result, Layer 7 is typically not as fast as the Layer 4 methods. Layer 7 is typically chosen when either enhanced options such as SSL termination, cookie based persistence, URL rewriting, header insertion/deletion etc. are required, or when the network topology prohibits the use of the layer 4 methods. The image below shows an example network diagram for this mode.



- Because layer 7 SNAT mode is a full proxy, Real Servers in the cluster can be on any accessible network including across the Internet or WAN.
- Layer 7 SNAT mode is not transparent by default, i.e. the Real Servers will not see the source IP address of the client, they will see the load balancer's own IP address by default, or any other local appliance IP address if preferred (e.g. the VIP address). This can be configured per layer 7 VIP. If required, the load balancer can be configured to provide the actual client IP address to the Real Servers in 2 ways. Either by inserting a header that contains the client's source IP address, or by modifying the Source Address field of the IP packets and replacing the IP address of the load balancer with the IP address of the client. For more information on these methods please refer to [Transparency at Layer 7](#).
- Layer 7 SNAT mode can be deployed using either a one-arm or two-arm configuration. For two-arm deployments, **eth1** is typically used for client side connections and **eth0** is used for Real Server connections, although this is not mandatory since any interface can be used for any purpose.

- Requires no mode-specific configuration changes to the load balanced Real Servers.
- Port translation is possible with Layer 7 SNAT mode, e.g. VIP:80 → RIP:8080 is supported.
- You should not use the same RIP:PORT combination for layer 7 SNAT mode VIPs and layer 4 SNAT mode VIPs because the required firewall rules conflict.

Note

For detailed configuration examples using various modes, please refer to [Chapter 10 - Configuration Examples](#).

Which Load Balancing Method Should I Use?

Mode Summary

Layer 4 DR Mode - This mode offers the best performance and requires limited physical Real Server changes. The load balanced application must be able to bind to the Real Server's own IP address and the VIP at the same time. This mode requires the "ARP Problem" to be solved, for more details please refer to [DR Mode Considerations](#). Layer 4 DR mode is transparent, i.e. the Real Servers will see the source IP address of the client.

Layer 4 NAT Mode - This mode is also a high performance solution but not as fast as DR mode. It requires the default gateway of each Real Server to be the load balancer and supports both one-arm and two-arm configurations. Layer 4 NAT mode is transparent, i.e. the Real Servers will see the source IP address of the client.

Layer 4 SNAT Mode - This mode is also a high performance solution but not as fast as the other layer 4 modes. It does not require any changes to the Real Servers and can be deployed in one-arm or two-arm mode. This mode is ideal for example when you want to load balance both TCP and UDP but you're unable to use DR mode or NAT mode due to network topology or Real Server related reasons. Layer 4 SNAT mode is non-transparent, i.e. the Real Servers will see the source IP address of the load balancer.

Layer 7 SNAT Mode - This mode offers greater flexibility but at lower performance levels. It supports HTTP cookie insertion, RDP cookies, Connection Broker integration and directly supports SSL termination or can forward traffic to STunnel or Pound if preferred. It also enables URL rewriting and header manipulation rules to be implemented. It does not require any changes to the Real Servers and can be deployed in either one-arm or two-arm mode. HAProxy is a high performance solution, but since it operates as a full proxy at layer 7 it cannot perform as fast as the layer 4 methods. Layer 7 SNAT mode is non-transparent by default, i.e. the Real Servers will see the source IP address of the load balancer rather than the client. This mode can be made transparent through the use of TProxy.

Our Recommendation

Where possible we recommend that Layer 4 Direct Routing (DR) mode is used. This mode offers the best possible performance since replies go directly from the Real Servers to the client, not via the load balancer. It's also relatively simple to implement. Ultimately, the final choice does depend on your specific requirements and infrastructure.

Note

If you are using Microsoft Windows Real Servers, we recommend that **NLB** (Network Load Balancing) is disabled on all servers to ensure that this does not interfere with the operation of the load balancer.



Chapter 4 - Appliance Fundamentals

Hardware Appliance Installation

1. Remove all packaging and rack mount the appliance if required.
2. Connect the power lead.



Note

The power supply is an auto-sensing unit (100v to 240v).

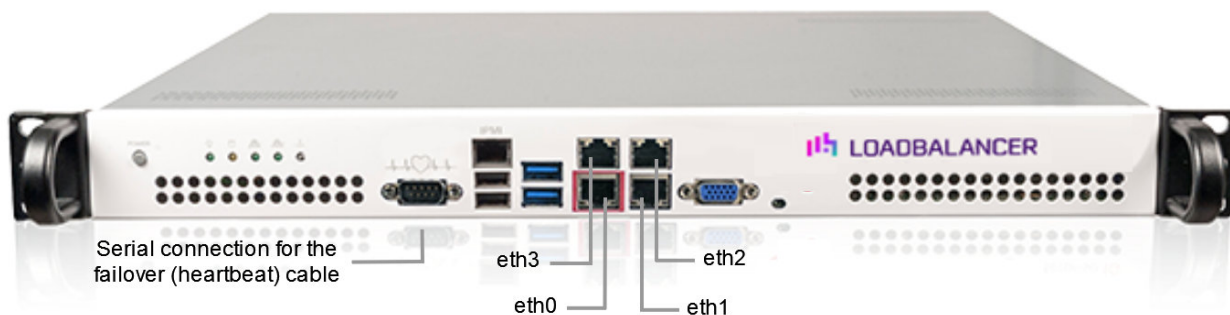
3. Connect a network cable from your switch to one of the Ethernet ports, typically **eth0** but this is not mandatory as each interface can be used for any purpose. If using a two-armed configuration connect another cable to a second Ethernet port, typically **eth1** but again this is not mandatory.
4. For a clustered hardware pair, the appliances must be able to communicate either via the network (ucast), via serial cable or both. By default, ucast only is used. If serial is preferred or you want to use both methods, connect a serial cable between the two appliances.



Note

If a serial cable is used, Heartbeat must be configured for this using the WebUI option: *Cluster Configuration > Heartbeat Configuration* and enabling 'Serial'.

5. Attach a monitor & keyboard to the appliance.
6. Check mains power is on and press the power switch. The fans should start & front panel LEDs should light.



Note

The above image shows the Enterprise Prime. For network interface information for other models please refer to [Front & Rear Panel Layouts](#).

7. Now follow the on-screen instructions to configure the management IP address and other network settings as detailed in [Configuring Initial Network Settings](#).

Virtual Appliance Installation

Supported Hypervisors

The VA is available for the following hypervisors:



- VMware vSphere/ESXi : v6.0 & later
- Virtual Box : v6.0 & later
- Microsoft Hyper-V : v2012 & later
- KVM : Kernel version v2.6.20 & later
- XEN : v6.0 & later
- Nutanix AHV

Host Requirements

To run the Loadbalancer.org Enterprise VA (irrespective of which Hypervisor is being used) the following basic server specifications must be met:

- 64bit CPU
- Virtual Technology hardware support - either Intel-VT-x or AMD-V compliant CPUs

Virtual Hardware Resource Allocations

By default the appliance is allocated the following resources:

- 2 vCPUs
- 4GB RAM
- 20GB disk

The CPU and memory allocations are suitable for a PoC or for low to medium throughput production applications. For more demanding situations, they can be increased as needed. Resources required depend on multiple factors including the application being load balanced, the number of end-users, the anticipated throughput, the underlying physical hardware running the hypervisor and whether you'll be load balancing at layer 4 or layer 7. Therefore it's not realistic to make generic recommendations. If you need assistance in determining the resources required for your deployment, please contact support.

Download & Extract the Appliance

1. Go to <https://www.loadbalancer.org/get-started/adcd-free-trial/>.
2. Enter your details (all fields except email address are optional).
3. Click the **Download** button.
4. Now click the **Download** link next to the appropriate hypervisor.
5. Unzip the contents of the file to your chosen location.

Note

All information provided is 100% confidential. If you're conducting a PoC (Proof of Concept) trial we may follow up with an email or phone call to see how you're getting on and offer assistance, but under no circumstances will Loadbalancer.org share your details with a third party.

Note

The same download is used for the licensed product, the only difference is that a license key file



(supplied by our sales team when the product is purchased) must be applied using the appliance's WebUI.

Note

The VA has 4 network adapters. For VMware, only the first adapter (**eth0**) is connected by default. For HyperV, KVM, XEN and Nutanix AHV all adapters are disconnected by default. Use the network configuration screen within the Hypervisor to connect the required adapters.

Hypervisor Deployment

VMware Host Client

1. Right-click **Host** in the VMware Host Client inventory and select **Create/Register VM**.
2. On the **Select creation type** page of the wizard, select **Deploy a virtual machine from an OVF or OVA file** and click **Next**.
3. On the **Select OVF and VMDK files** page, provide a unique name for the virtual machine.
4. Click the blue pane to open your local system storage, browse to the VA download location and select both the **.ovf** and **.vmdk** files.
5. Complete the remaining options according to your requirements and deploy the VA.

VMware vSphere Client

The steps below apply to VMware ESX/ESXi & vSphere Client v6.7 and later.

Upgrading to the latest Hardware Version

When the appliance is deployed, the virtual hardware version is set to 11. This enables compatibility with ESX version 6.0 and later. You can upgrade to a later hardware version if required.

Note

Create a snapshot or backup of the virtual machine first before upgrading.

Installing the Appliance using vSphere Client

1. Right-click the inventory object where the appliance is to be located and select **Deploy OVF Template**.
2. In the **Select an OVF Template** screen, select the **Local File** option, click **Browse**, navigate to the download location, select both the **.ovf** and **.vmdk** files and click **Next**.
3. In the **Select a name and folder** screen, type a suitable name for the appliance - this can be up to 80 characters in length.
4. Select the required location for the appliance - by default this will be the location of the inventory object from where the wizard was started and click **Next**.
5. In the **Select a compute resource** screen, select the required compute resource for the appliance - by default this will be the inventory object from where the wizard was started and click **Next**.
6. In the **Review details** screen, verify the template details and click **Next**.
7. In the **Configuration** screen, select the required CPU/RAM deployment configuration and click **Next**.

Note

These settings can be changed after deployment if needed.



8. In the **Select Storage** screen, first select the required storage location for the appliance.
9. Now select the required disk format and click **Next**.

 **Note**

Loadbalancer.org recommends selecting a thick provision format. By default the appliance disk is 20GB.

10. In the **Select Networks** screen, select the required destination network using the drop-down next to **VM Network** and click **Next**.
11. In the **Ready to complete** screen, review the settings and click **Finish** to create the virtual appliance. To change a setting, use the **Back** button to navigate back through the screens as required.

Configure Additional Network Adapters

The appliance has 4 network adapters. By default only the first adapter is connected. This will be **eth0** when viewed in the appliance WebUI. If you require additional network adapters to be connected, follow the steps below:

1. Right-click the appliance, select **Edit Settings**.
2. Using the drop-down next to the relevant network adapter(s), select the required **Network** and tick (check) the **Connected** check-box and click **OK**.

Start the Appliance

Now power up the appliance.

VMware Workstation Player

1. Select **Player > File > Open**.
2. Browse to the VA download location and select the **.ovf** file.
3. Modify the default name as required and click **Import**.

VMware Tools / Open VM Tools

From v8.8.1 all new VAs use Open VM Tools rather than VMware Tools.

When updating a v8.7.x appliance to v8.8.1 using online or offline update, VMware tools is removed and Open VM Tools is installed.

To Verify that Open VM Tools is running

Run the following command at the console or via an SSH session:

```
# service vmtoolsd status
vmtoolsd (pid 3514) is running...
```

 **Important**

When using open-vm-tools, the VMware Tools status is shown as "Guest Managed" on the virtual machine Summary tab. The status Guest Managed means that you cannot use the vCenter Server to manage VMware Tools and you cannot use vSphere Update Manager to

Microsoft Hyper-V

The steps below apply to Windows Hyper-V 2012 & later.

Installing the Appliance using Hyper-V Manager

1. Start Hyper-V Manager, then using the right-click menu or the Actions pane select **Import Virtual Machine** and click **Next**.
2. In the **Locate folder** screen, browse to the location of the extracted download and select the **Loadbalancer.org Ent. VA** folder.
3. Click **Next** until you reach the **Choose Import Type** screen, select the option **Copy the virtual machine (create a new unique ID)** and click **Next**.
4. In the **Choose Folder For Virtual Machine Files** screen, tick (check) the checkbox **Store the Virtual Machine in different location**, then select a suitable location for the virtual machines files and click **Next**.
5. In the **Choose Folder to store Virtual Hard Disks** screen, select a suitable location for the virtual hard disk files and click **Next**.
6. In the **Completing Import Wizard** screen, verify that all settings are correct and click **Finish** to complete the import process. To change a setting, use the **Previous** button to navigate back through the screens as required.

Once complete, the load balancer will appear in the **Virtual Machines** list.



Note

For a clustered pair, make sure that you select a different folder location in steps 4 & 5.

Configure Network Adapters

The appliance has 4 network adapters, these remain disconnected once deployment completes. To connect an adapter to a virtual switch:

1. Right-click the appliance, select **Settings**.
2. Select the first network adapter and set the required virtual switch that the adapter should be connected to. This will be **eth0** when viewed in the appliance WebUI.
3. Click **Apply**.



Note

If you need to connect additional network adapters repeat step 2 as required.

Start the Appliance

Now power up the appliance.

Linux Integration Services

Linux Integration Services are pre-installed by default. Therefore manual installation is not required.

KVM



The following steps should be followed on the KVM host:

1. Extract the archive to **/var/lib/libvirt/images/**.
2. `virsh define Loadbalancer*.xml`.
3. `virsh start Loadbalancer*`.

Note

Network cards are set to NAT by default so adjust as needed before powering on. Please also refer to the XML file for additional configuration notes.

Nutanix AHV

For detailed installation and deployment guidance, please refer to our [Nutanix blog](#).

XEN

The following steps should be followed on the XEN host:

1. Extract the archive.
2. Import the **xva** file into XEN.


Cloud Appliance Installation

For details of our cloud based products, please refer to the relevant guide in the [documentation library](#).

Configuring Initial Network Settings

After power up, the following startup message is displayed on the appliance console:

```

 Welcome to the Loadbalancer.org appliance.

To perform initial network configuration, log in to the console as
Username: setup
Password: setup

To access the web interface and wizard, point your browser at
http://192.168.2.21:9080/
or
https://192.168.2.21:9443/

lbmaster login:
```

As mentioned in the text, to perform initial network configuration, login as the "setup" user at the appliance console.

Once logged in, the Network Setup Wizard will start automatically. This will enable you to configure the management IP address and other network settings for the appliance.

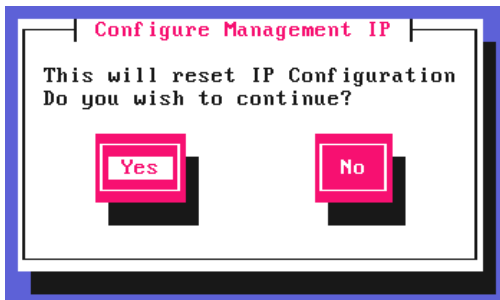
login to the console:

Username: setup

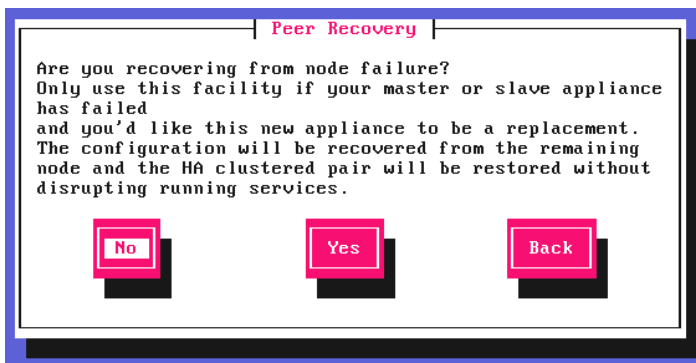
Password: setup

A series of screens will be displayed that allow network settings to be configured:

In the **Configure Management IP** screen, leave **Yes** selected and hit <ENTER> to continue.



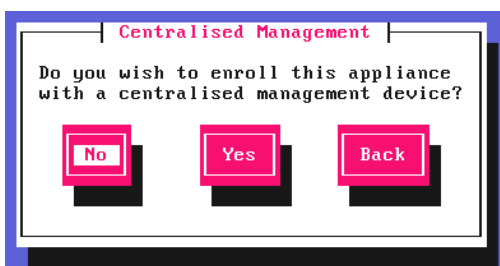
In the **Peer Recovery** screen, leave **No** selected and hit <ENTER> to continue.



Note

For more details on node recovery using this option please refer to [Disaster Recovery After Node \(Primary or Secondary\) Failure](#).

In the **Centralized Management** screen, if you would like to enroll the appliance with a management server (typically portal.loadbalancer.org), select **Yes**, otherwise leave **No** selected, then hit <ENTER> to continue. If you select **Yes**, you'll be asked to confirm the server's details and provide login credentials at the end of this setup process.

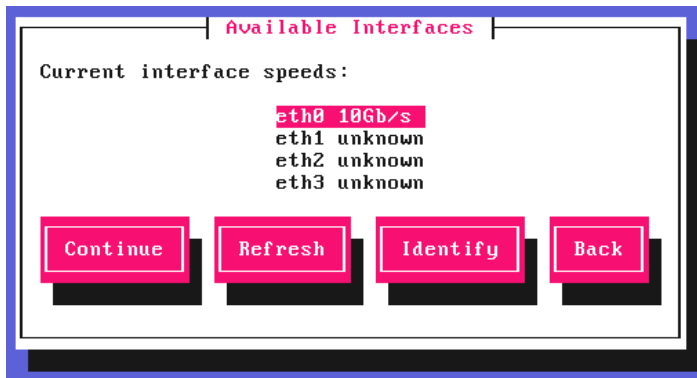


Note

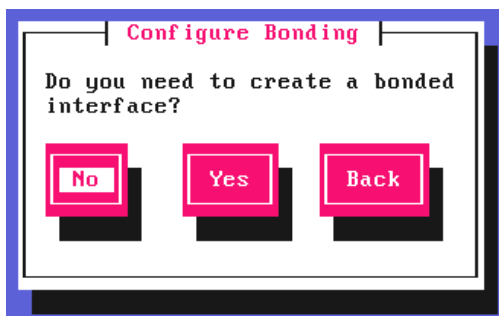
For information on how to modify Centralized Management settings via the WebUI, please refer to [Portal Management & Appliance Adoption](#).

In the **Available Interfaces** screen, a list of available interfaces will be displayed, hit <ENTER> to continue.

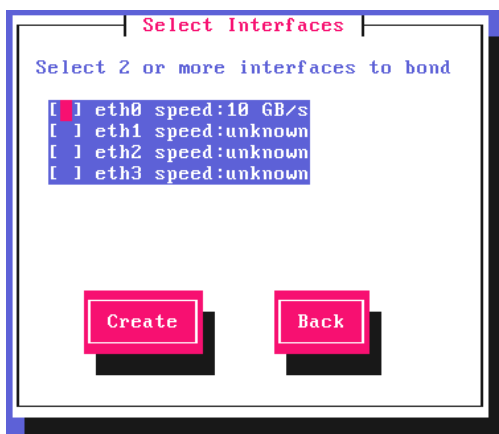




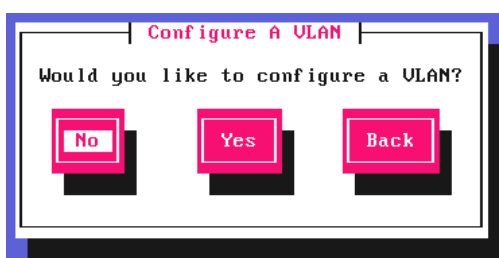
In the **Configure Bonding** screen, select **Yes** if you want to configure a bonded interface, if not leave **No** selected, then hit <ENTER> to continue.



If you select **Yes**, the **Select Interfaces** screen will be displayed. Using the space bar, select the interfaces you'd like to include in the bond, select **Create** and hit <ENTER> to continue.

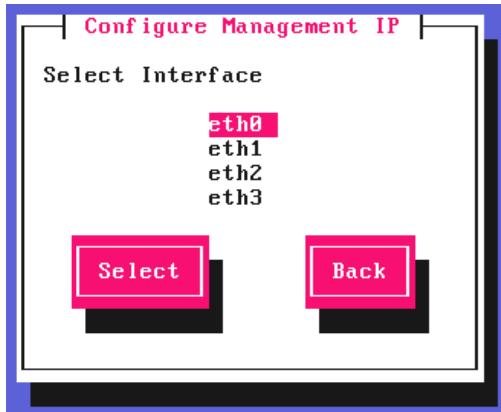


In the **Configure a VLAN** screen, select **Yes** if you want to configure a VLAN, if not leave **No** selected, then hit <ENTER> to continue.




If you select **Yes** you'll be prompted to enter a VLAN Tag ID.

In the **Configure Management IP** screen, select the interface that'll be used to manage the appliance, then hit <ENTER> to continue.



```
Configure Management IP
Select Interface
eth0
eth1
eth2
eth3
Select Back
```

In the **Set IP address** screen, either enter the required *Static IP Address & CIDR Prefix* and select **Done** or select **Use DHCP**, then hit <ENTER> to continue.

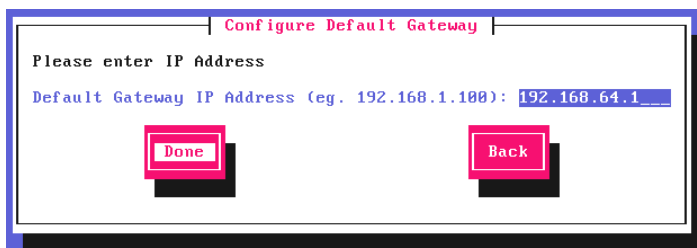


```
Set IP Address for eth0
Please enter IP and CIDR Subnet Mask
Static IP Address (eg. 192.168.1.100): 192.168.111.150
CIDR Prefix (eg. 24): 18
Done Use DHCP Back
```

Note

A subnet mask such as 255.255.255.0 is not valid, in this case enter 24 instead.

In the **Configure Default Gateway** screen, enter the required *Default Gateway IP Address*, select **Done** and hit <ENTER> to continue.



```
Configure Default Gateway
Please enter IP Address
Default Gateway IP Address (eg. 192.168.1.100): 192.168.64.1
Done Back
```

In the **Configure DNS Servers** screen, configure the required DNS server(s), select **Done** and hit <ENTER> to continue.

Configure DNS Servers

Please enter IP Address

Primary DNS Server (eg. 192.168.1.100): 3.8.8.8

Secondary DNS Server (Leave blank to omit):

Done Back

In the **Set Password** screen, hit <ENTER> to continue.

Set Password

Please set a password. This password will be used for the WUI and the root console. NOTE: You will not be able to access the console until it is enabled from the WUI.

OK

Enter the *Password* you'd like to use for the **loadbalancer** WebUI user account and the **root** Linux user account. Repeat the password, select **Done** and hit <ENTER> to continue.

Set Password

Please set a password for the WebUI and the root console:

Password: *****

Password Again: *****

Done Back

If you selected **Yes** when asked if you want to enroll for Centralized Management, you'll now be prompted for the details. Default values for the *Host* and *Port* are set and can be changed if required. Enter the *Username* and *Password* for the management server account you'd like the appliance to be associated with, select **Done** and hit <ENTER> to continue.

Enroll in Centralised Management

Please enter the details for the Centralised Management Device to enroll with:

Host: portal.loadbalancer.org

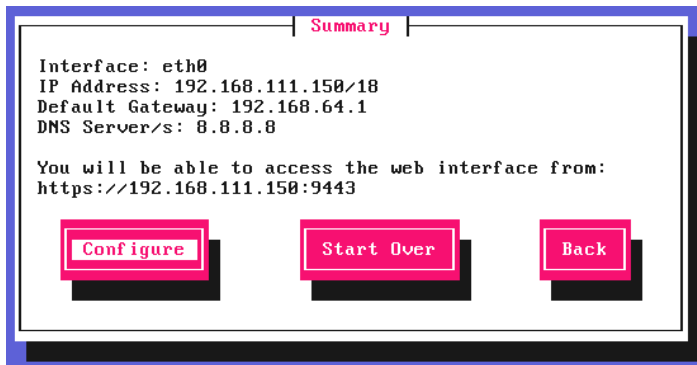
Port: 443

Username:

Password:

Done Back

In the **Summary** screen, check all settings. If everything is correct, leave **Configure** selected and hit <ENTER> to continue. All settings will be applied. If you need to change a setting, use the **Back** button.



Once the configuration has been written, the **Configuration Complete** screen and message will be displayed. Click **OK** to exit the wizard and return to the command prompt.



Appliance Access & Configuration Methods

Once the initial network settings have been configured, the appliance can be accessed and configured using the Web User Interface (WebUI). In addition, full "root" user access is provided.

Accessing the Appliance WebUI

The WebUI is accessed using a web browser. By default, users are authenticated using Apache authentication. Users can also be authenticated against LDAP, LDAPS, Active Directory or Radius - for more information, please refer to [External Authentication](#).

Note

There are certain differences when accessing the WebUI for the cloud appliances. For details, please refer to the relevant [Quick Start / Configuration Guide](#).

1. Using a browser, navigate to the following URL:

`https://<IP-address-configured-during-the-network-setup-wizard>:9443/lbadmin/`

Note

You'll receive a warning about the WebUI's SSL certificate. This is due to the default self signed certificate that is used. If preferred, you can upload your own certificate - for more information, please refer to [Appliance Security Features](#).

Note

If you need to change the port, IP address or protocol that the WebUI listens on, please refer to [Service Socket Addresses](#).

2. Log in to the WebUI using the following credentials:

Username: loadbalancer

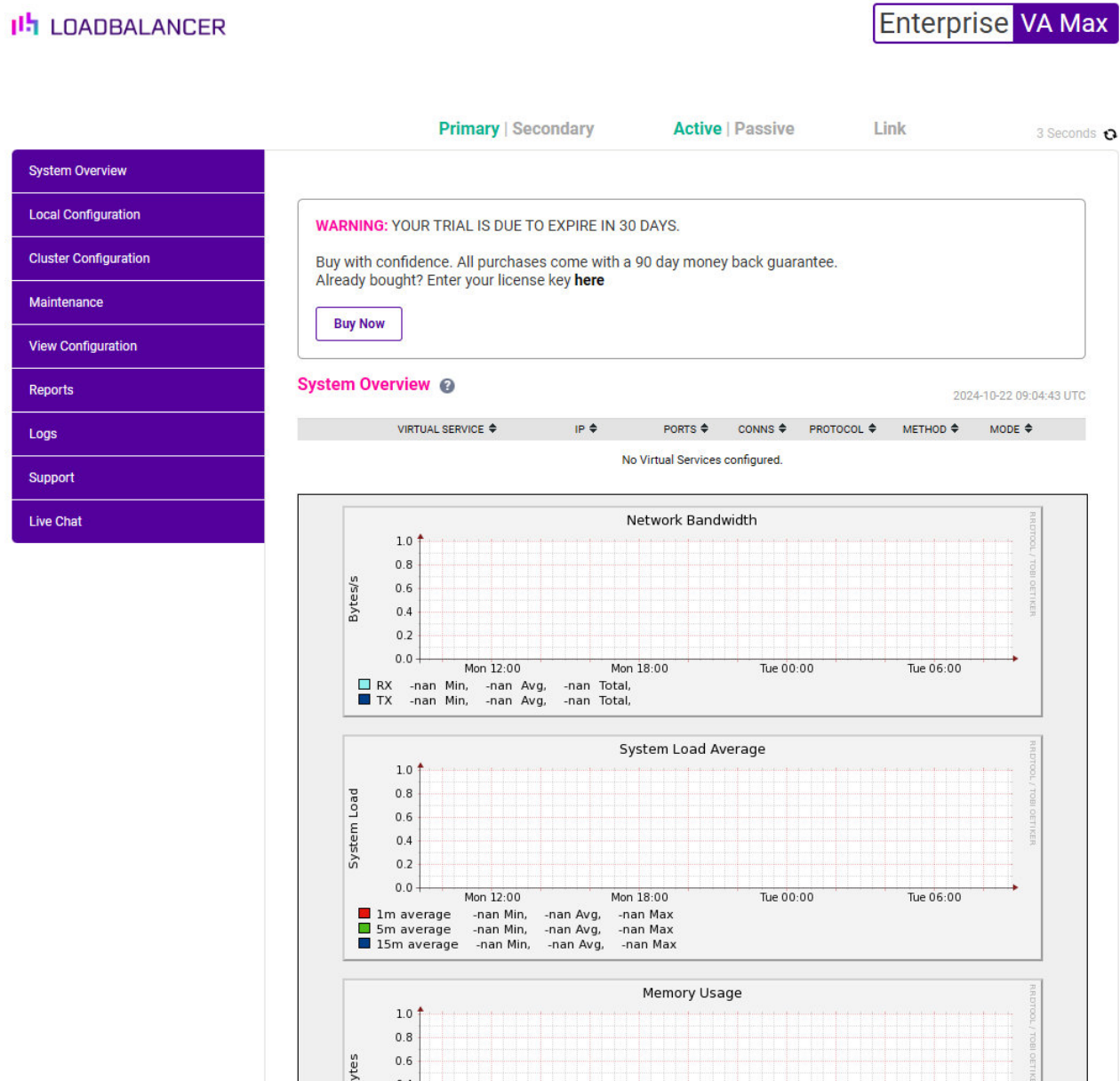
Password: <configured-during-network-setup-wizard>



Note

To change the password, use the WebUI menu option: **Maintenance > Passwords**.

Once logged in, the WebUI will be displayed as shown below:



Main Menu Options

System Overview - Displays a graphical summary of all VIPs, RIPs and key appliance statistics

Local Configuration - Configure local host settings such as IP address, DNS, system time etc.

Cluster Configuration - Configure load balanced services such as VIPs & RIPs

Maintenance - Perform maintenance tasks such as service restarts and creating backups

View Configuration - Display the saved appliance configuration settings

Reports - View various appliance reports & graphs



Logs - View various appliance logs

Support - Create a support download, contact the support team & access useful links

Live Chat - Start a live chat session with one of our Support Engineers

 **Note**

If you've already purchased a license, please refer to [Installing the License Key](#).

"Root" User Access

Log in to the console or via an SSH session:

Username: root

Password: <configured-during-network-setup-wizard>

 **Note**

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: **Local Configuration > Security**. You'll need to Set **Appliance Security Mode** to **Custom**, configure the required option(s) and click **Update**.

 **Note**

For Windows hosts [PuTTY](#) can be used for SSH access and [WinSCP](#) can be used for secure file transfer.

 **Note**

If you need to change the port or IP address that SSH listens on, please refer to [Service Socket Addresses](#). The allowed ciphers and other advanced SSH settings can be configured using the WebUI menu option **Local Configuration > Physical - Advanced Configuration** and scrolling to the **SSH Service** section.

Keyboard Layout

By default the appliance is configured with a US keyboard layout. The layout can be changed by editing the file **/etc/sysconfig/keyboard**. For example, to change the layout from US to UK:

1. edit **/etc/sysconfig/keyboard** using a text editor such as **vi** or **vim** for Linux or WinSCP under Windows.
2. replace KEYTABLE="us" with KEYTABLE="uk".
3. replace Layout="us" with Layout="uk".
4. save the file and re-boot the appliance.

Chapter 5 - Appliance Management

Installing the License Key

The appliance can be used completely unrestricted for 30 days without installing a license key. After 30 days, the appliance continues to work but it's no longer possible to make configuration changes.

Note

if you're conducting a PoC (Proof of Concept) using the VA and require more time to complete your evaluation, please contact sales@loadbalancer.org who will be able to provide guidance on how to extend the trial.

For an unlicensed VA, the following message is displayed:

WARNING: YOUR TRIAL IS DUE TO EXPIRE IN 30 DAYS.

Buy with confidence. All purchases come with a 90 day money back guarantee.
Already bought? Enter your license key **here**

[Buy Now](#)

For an unlicensed hardware appliance, the following message is displayed:

WARNING: This appliance is unregistered. **Please enter your license key** within 30 days to activate your appliance.
If you do not have your license key please **Contact Us**

To install the license key:

1. Using the WebUI, navigate to: *Local Configuration > License Key*.

Install License Key

This unit is in evaluation mode. Please enter your license key to remove this restriction.

If you do not have a license key, please contact sales@loadbalancer.org.

No file chosen

2. Click **Choose File** then browse to and select the license file provided when the appliance was purchased.
3. Click **Install License Key**.

Note

Once the license is applied, these warning messages will no longer be displayed.



Appliance Software Update

We recommend that the appliance is kept up to date to ensure that you benefit from the latest bug fixes, security updates and feature improvements. Both online and offline update are supported.

Note

Services may need to be restarted/reloaded after the update process completes or in some cases a full appliance restart may be required. We therefore recommend performing the update during a maintenance window.

Determining the Current Software Version

The software version is displayed at the bottom of the WebUI as shown in the example below:

Copyright © Loadbalancer.org Inc. 2002 – 2024

ENTERPRISE VA Max - v8.12.2

English ▾

Creating a Backup / Checkpoint

We recommend that a backup or checkpoint (v8.11.0 and later) is created before running the update. This can be done using the WebUI option: **Maintenance > Backup & Restore** and selecting the **Backup** or **Checkpoints** tab. For more information on creating backups and checkpoints, please refer to [Backup & Restore](#) and [Disaster Recovery](#).

Online Update

By default, the appliance periodically contacts the Loadbalancer.org update server (**update.loadbalancer.org**) and checks for updates. This behavior can be disabled if preferred as detailed below.

Configuring Online Update

To disable auto-check for updates:

1. Using the WebUI, navigate to: **Local Configuration > Physical - Advanced Configuration**.
2. Scroll down to the **Online Updates** section.

Online Updates	
Update Server	<input type="text" value="update.loadbalancer.org"/> ?
Auto-check for updates	<input type="checkbox"/> ?

3. Disable (uncheck) **Auto-check for updates**.
4. Click **Update**.

Manual Check for Updates

If auto-check for updates is disabled, the update check must be initiated manually.



To initiate an online update check:

1. Using the WebUI, navigate to: **Maintenance > Software Update**.
2. Click **Online Update**.
 - If no update is available, a message similar to the following will be displayed:

Information: Version v8.12.2 is the current release. No updates are available

- If an update is available, click **Online Update** and proceed as detailed below.

Auto-Check for Updates

if **Auto-check for updates** is left at its default value and an update is found when the next scheduled check occurs, a notification similar to the example below will be displayed at the top of the WebUI:

Information: Update 8.13.0 is now available for this appliance.

Online Update

To perform the update:

1. Click **Online Update**.
2. A summary of all new features, improvements, bug fixes and security updates included in the update will be displayed.
3. Click **Update** at the bottom of the page to start the update process.

(!) Important Do not navigate away whilst the update is ongoing, this may cause the update to fail.

The update can take several minutes depending on download speed and upgrade version. Once complete, the following message will be displayed:

Information: Update completed successfully. Return to **system overview**.

If services need to be reloaded/restarted or the appliance needs a full restart, you'll be prompted accordingly.

Offline Update

If the appliance does not have access to the Internet, offline update can be used.

To check for the latest version, please refer to our product roadmap page available [here](#). To obtain the latest offline update files contact support@loadbalancer.org.



To perform an offline update:

1. Using the WebUI, navigate to: **Maintenance > Software Update**.
2. Select **Offline Update**.
3. The following screen will be displayed:

Software Update

Offline Update

The following steps will lead you through offline update.

1. Contact **Loadbalancer.org support** to obtain the offline update archive and checksum.
2. Save the archive and checksum to your local machine.
3. Select the archive and checksum files in the upload form below.
4. Click *Upload and Install* to begin the update process.

Archive: No file chosen

Checksum: No file chosen

4. Select the *Archive* and *Checksum* files.
5. Click **Upload and Install**.
6. If services need to be reloaded/restarted or the appliance needs a full restart, you'll be prompted accordingly.

Updating a Clustered Pair

Note

Since services may need to be reloaded or restarted during the update process, we recommend performing the update during a maintenance window.

To update a clustered pair:

1. Perform the update on the Secondary (passive) appliance first.
2. Restart/reload services or reboot the appliance as directed.
3. Failover to the Secondary so this is now the active appliance.
4. Now update the Primary appliance in the same way.
5. Restart/reload services or reboot the appliance as directed.
6. Failback to the Primary so this is the active appliance.

Note

For a clustered pair, we recommend fully testing & validating the Primary/Secondary failover process before going live. For more information, please refer to [Testing & Verifying Primary/Secondary Replication & Failover](#).



Network Configuration

Physical/Virtual Adapters

The Enterprise Prime, Enterprise Flex and all virtual models have 4 network adapters. The Enterprise Max has either 4 or 6 adapters depending on the chosen configuration when purchased. Comprehensive information on all models is available on our [website](#).

If multiple logical interfaces are required, these can be added by specifying multiple IP addresses as shown below. If additional network connectivity is required, an external switch can be used.

Typically, the main reason for using all 4 or 6 interfaces is when bonding (e.g. 802.3ad) is required in a two-arm NAT or SNAT mode highly available configuration.

Multiple VLANs can be configured if required. For details, please refer to [Configuring VLANs](#).

Note

For the VMware appliance, only the first adapter (**eth0**) is connected by default. For HyperV, KVM, XEN and Nutanix AHV all adapters are disconnected by default. Use the network configuration screen within the Hypervisor to connect the required adapters.

Configuring IP Addresses

Initial network settings including the management IP address are configured using the Network Setup Wizard. For more information, please refer to [Configuring Initial Network Settings](#).

Once the management address is configured, additional IP addresses can be configured using the WebUI menu option: **Local Configuration > Network Interface Configuration**.

For one-arm deployments, **eth0** is normally used. For two-arm deployments, **eth0** is typically used as the internal interface and **eth1** is used as the external interface. This is not mandatory and each interface can be used for any purpose.

CIDR notation is used to specify IP addresses and subnet masks. For example, to specify an IP address of **192.168.2.100** with a subnet mask of **255.255.255.0**, **192.168.2.100/24** would be entered in the relevant interface field as shown in the example below:

eth0

192.168.2.100/24

Note


For information on CIDR notation, please refer to [Appliance IPv4 Address Format \(CIDR notation\)](#).


To configure IP address(es):





1. Using the WebUI, navigate to: *Local Configuration > Network Interface Configuration*.
2. Configure the required IP address/mask. Multiple addresses can be assigned as shown below:

IP Address Assignment


eth0


eth1


eth2


eth3

eth0	192.168.10.100/24	MTU <input style="width: 50px; text-align: center;" type="text" value="1500"/> bytes
eth1	192.168.20.100/24 192.168.40.100/24	MTU <input style="width: 50px; text-align: center;" type="text" value="1500"/> bytes
eth2		MTU <input style="width: 50px; text-align: center;" type="text" value="1500"/> bytes
eth3		MTU <input style="width: 50px; text-align: center;" type="text" value="1500"/> bytes

3. Click **Configure Interfaces**.

Configuring Bonding

Multiple network adapters can be bonded.

To configure bonding:

1. Using the WebUI, navigate to: *Local Configuration > Network Interface Configuration*.
2. Any of the available adapters can be bonded. For example, to bond **eth0** and **eth1**, select (check) the **eth0** and **eth1** checkboxes as shown below:

Create New Bond		
eth0:	<input checked="" type="checkbox"/>	Interface Speed:
eth1:	<input checked="" type="checkbox"/>	Interface Speed: Link Down
eth2:	<input type="checkbox"/>	Interface Speed: Link Down
eth3:	<input type="checkbox"/>	Interface Speed: Link Down
Bonding Mode	Mode 1 - (Default) Active/Backup ?	
		Create

3. Select the required **Bonding Mode**:

Mode 0 - Balance round robin. Transmits packets in a numerical order from the first available Secondary through to the last.

Mode 1 - Active Backup (default). This places one of the adapters in a backup state and will only become active if the link is lost to the active adapter. This mode provides fault tolerance.

Mode 4 - 802.3ad. Dynamic link aggregation mode. This mode requires a switch that supports IEEE 802.3ad.

4. Click **Create**.

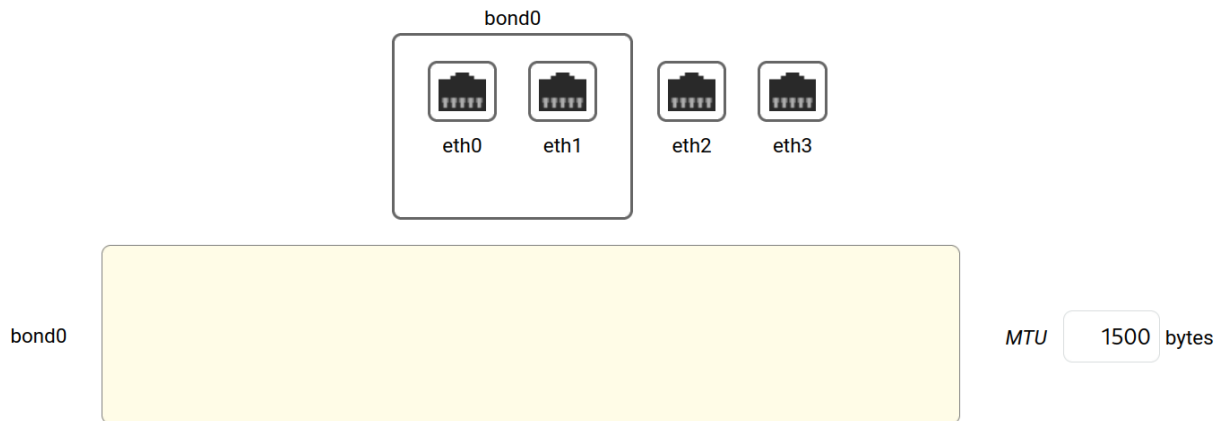
5. The new bond (**bond0**) is displayed as shown below:

Create New Bond		
eth2:	<input type="checkbox"/>	Interface Speed: Link Down
eth3:	<input type="checkbox"/>	Interface Speed: Link Down
Bonding Mode	Mode 1 - (Default) Active/Backup ?	
		Create

Active Bonds		
bond0	Mode 1	Interfaces: eth0,eth1 Delete

Note

At this point the adapters still have the same IP settings configured previously. Once an IP address is configured for the bond and **Configure Interfaces** is clicked, these addresses will be removed and only the bond address will apply. If the bond is deleted, these addresses will be re-applied to the adapter(s).



6. Enter the IP address for **bond0** and click **Configure Interfaces**.
7. If this is a new bonding configuration or the bonding mode has been changed, restart the appliance using the WebUI menu option: **Maintenance > System Control** and clicking **Restart Load Balancer**.

⚠ Important

For a clustered pair you'll need to configure bonding in the same way on both appliances. Failure to do so will result in heartbeat communication issues.

Configuring VLANs

Native 802.1Q VLAN support can be configured if required.

In **access mode** switch ports are dedicated to one VLAN. The switch handles all the tagging and de-tagging of frames - the station connected to the port does not need to be configured for the VLAN at all.






In **trunk mode** the switch passes on the raw VLAN frames - the station connected must be configured to handle them. Trunk mode is usually used to connect two VLAN-carrying switches, or to connect a server or router to a switch.

If the load balancer is connected to an access mode switch port, no VLAN configuration is required. If the load balancer is connected to a trunk port, then all the required VLANs must be configured on the load balancer.

To configure a VLAN:

1. Using the WebUI, navigate to: **Local Configuration > Network Interface Configuration**.
2. In the VLAN section select the required adapter (e.g. **eth0**).
3. Enter the VLAN ID (e.g. **250**).
4. Click **Add VLAN**.
5. An extra IP Address Assignment field named **eth0.250** will be created, enter the required IP address here.

IP Address Assignment

					
	eth0	eth0.250	eth1	eth2	eth3
eth0	<div>192.168.111.220/18</div>				
				MTU 1500 bytes	
eth0.250	<div></div>				
				MTU 1500 bytes	
				<button>Delete eth0.250</button>	

6. Click **Configure Interfaces**.

7. To delete the VLAN definition, click the appropriate **Delete** button.

(!) Important For a clustered pair you must configure VLANs in the same way on both appliances.

Interface Offloading

This will enable (when supported) hardware NIC offloading. This option is enabled by default.

To configure Interface offloading:

1. Using the WebUI, navigate to: *Local Configuration > Physical - Advanced Configuration*.
2. Scroll down to the *Interface Offload* section.
3. configure the required setting.
4. Click **Update**.

Configuring MTU Settings

To set the MTU for an interface:

1. Using the WebUI, navigate to: *Local Configuration > Network Interface Configuration*

eth0	<div>192.168.10.100/24</div>	MTU 1500 bytes
------	------------------------------	----------------

2. Enter the required *MTU* setting - the default is **1500**.

3. Click **Configure Interfaces**.

Configuring Default Gateway & Static Routes

To set the default gateway for IPv4 and IPv6:

1. Using the WebUI, navigate to: *Local Configuration > Routing*.
2. Specify the required **Default Gateway** as shown in the example below:

Routing

Default Gateway			
IP v4	<input type="text" value="192.168.1.254"/>	via interface	<input type="text" value="auto"/> ?
IP v6	<input type="text"/>	via interface	<input type="text" value="auto"/> ?

3. Click **Configure Routing**.

To configure static routes:

1. Using the WebUI, navigate to: *Local Configuration > Routing*.
2. Specify the required **Static Routes** as shown in the example below:

Static Routes			
Subnet	<input type="text" value="10.10.0.0/16"/>	via gateway	<input type="text" value="10.10.1.254"/>
Subnet	<input type="text" value="10.20.0.0/16"/>	via gateway	<input type="text" value="10.20.1.254"/>
Subnet	<input type="text"/>	via gateway	<input type="text"/>

3. Click **Configure Routing**.

Note

Unlimited static routes can be configured. Additional rows will be automatically added to the WebUI screen as needed.

Management Gateway

If you want to manage the appliance from a remote subnet that is accessible via a different gateway, the relevant management address and associated gateway must be configured. Traffic from the management address is then routed via the management gateway rather than via the default gateway.

Note

This is not a security feature, it's a routing option. By default, the WebUI is still accessible on all appliance IPs as before (unless this has been changed - for more details, please refer to [Service Socket Addresses](#)). Policy based routing is used to provide this feature. For more information, please refer to the "PBR" section below.



To configure the management gateway:

1. Using the WebUI, navigate to: *Local Configuration > Physical Advanced Configuration*.
2. Scroll down to the *Management Gateway* section.

Management Gateway	
Management Address	<div>none ▾</div> ?
Via Gateway	<div></div> ?

3. Select the required *Management Address* from the dropdown.



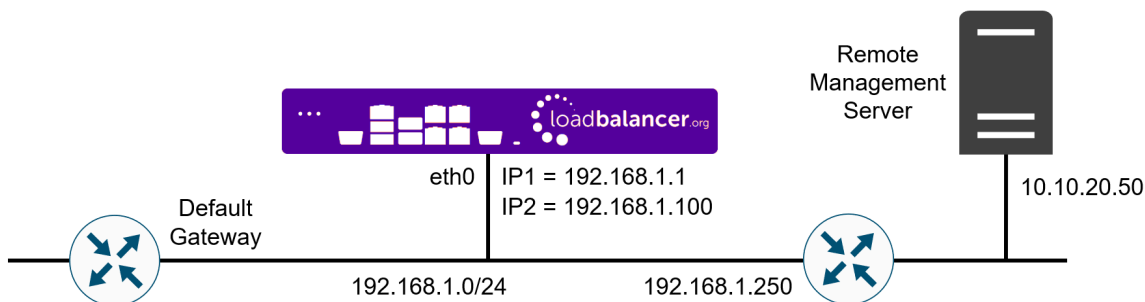
Note

The dropdown is populated with all IP addresses that have been assigned to the various network interfaces.

4. Specify the required *Gateway* address.
5. Click **Update**.

Configuration Example

To enable appliance management on IP **192.168.1.100** from the remote management server shown below:



The following *Management Address* and *Via Gateway* settings would be required:

Management Gateway	
Management Address	<div>192.168.1.100 ▾</div> ?
Via Gateway	<div>192.168.1.250</div> ?



Note

You'll need to ensure that **192.168.1.100** is configured on **eth0**. This can be done using the WebUI menu option: *Local Configuration > Network Interface Configuration*.

Policy Based Routing (PBR)

If you require a custom gateway for a particular VIP, this can be achieved using PBR.



Tip

If client source addresses are known and predictable, static routes should normally be used to route traffic. In other situations where this is not known or the network is large with many subnets, PBR can be used. Here, return traffic is routed based on the source address of the reply traffic (the VIP/floating IP) rather than on the destination address (the client's IP).

To configure a VIP to return traffic via a custom gateway rather than via the default gateway:

1. Using the WebUI, navigate to: *Cluster Configuration > PBR Default Gateways*.

Add PBR Default Gateway

Select Floating IP	192.168.111.130 ▼
Local only	<input checked="" type="checkbox"/>
Gateway Address	192.168.111.254
<div>Submit</div>	

PBR Default Gateways

Floating IP	Gateway IP	Local	Table Name
-------------	------------	-------	------------

2. Set the *Floating IP* to the required VIP address, e.g. **192.168.111.130**.
3. Configure *Local Only* according to your requirements. When enabled, only the link local route for the VIP is set, when not enabled, all link local routes are set.
4. Set the *Gateway Address* to the required value, e.g. **192.168.111.254**.
5. Click **Submit**.
6. Once configured, the new default PBR gateway will be displayed as shown below:

PBR Default Gateways

Floating IP	Gateway IP	Local	Table Name	
192.168.111.130	192.168.111.254	local	lbpbr-2	<div>Delete</div>

To delete the new gateway and use the appliance's standard default gateway, click **Delete**.

Configuring Hostname & DNS

To set the hostname, domain & DNS servers:

1. Using the WebUI, navigate to: *Local Configuration > Hostname & DNS*.



Hostname & DNS

Hostname	<input type="text" value="lbmaster"/>	?
Domain Name	<input type="text" value="localhost"/>	?
Domain Name Server	Primary <input type="text" value="8.8.8.8"/>	?
	Secondary <input type="text"/>	?
	Tertiary <input type="text"/>	?

Update

2. Specify the required *Hostname*, by default this is set to **lbmaster**.
3. Specify the required *Domain Name*, by default this is set to **localhost**.
4. Specify the required *Domain Name Server(s)*.
5. Click **Update**.

Service Socket Addresses

By default, the WebUI, SSH, GSLB, SNMP, Gateway, Shuttle and the fallback Server are bound to the IPs, ports and protocols shown in the table below.

Service	Default IP Address	Default Port	Protocol
WebUI	* (all IPs)	9080	HTTP
WebUI	* (all IPs)	9443	HTTPS
SSH	* (all IPs)	22	TCP
GSLB	* (all IPs)	53	TCP & UDP
SNMP	* (all IPs)	161	TCP
SNMP	* (all IPs)	161	UDP
Gateway	Management IP address	9000	TCP
Shuttle	Management IP address	25565	TCP
Fallback Server	* (all IPs)	9081	TCP

Service sockets can be configured to listen on all IP's, the IPv4 loopback address (127.0.0.1), the IPv6 loopback address (::1) or set to a specific interface address. The default setting for each is shown in the table above.

To configure service socket addresses:

1. Using the WebUI, navigate to: **Local Configuration > Physical - Advanced Configuration**.
2. Scroll to the **Service Socket Addresses** section.



Service Socket Addresses					
WebUI	https	*	9079	Delete	?
	https	*	9443	Delete Add	
SSH	tcp	*	22	Delete Add	?
GSLB	tcp+udp	*	53	Delete Add	?
SNMP	tcp	*	161	Delete	?
	udp	*	161	Delete Add	
Gateway Service	tcp	192.168.120.50	9000	Delete Add	?
Shuttle	tcp	192.168.120.50	25565	Delete Add	?
Fallback Server	http	*	9081	Delete Add	?

3. Configure the required protocols, IP addresses and ports. Service sockets can be added or deleted for all services except the Gateway and Shuttle using the buttons provided.
4. Click **Update**.
5. Restart the relevant service(s) using the button(s) in the "Commit changes" message box at the top of the screen.

Appliance Security Features

The appliance has a range of security related features that can be used to help ensure the appliance is secure.

Security Mode

To control how the appliance is accessed and which features are enabled, three security modes are provided:

- **Custom** - In this mode the following security options can be configured to suit your requirements:
 - **Disable Console Access** - Disable root user console access
 - **Disable SSH Password Access** - Disable SSH password access, users must use SSH keys to login
 - **Web User Interface via HTTPS only** - WebUI connections are forced to use HTTPS, connections are redirected to the port specified in the *HTTPS Port for Web User Interface* field
- **Secure - (Default)** - In this mode:
 - "root" user console access & SSH password access are disabled
 - WebUI connections are forced to use HTTPS
 - Access to the *Local Configuration > Execute shell command* menu option is disabled
 - The Firewall Script & the Firewall Lockdown Wizard Script cannot be edited

- **Secure - Permanent** - This mode is the same as **Secure** but once set it cannot be changed



Warning

Setting the security mode to **Secure - Permanent** is irreversible.

To configure the security mode & related options:

1. Using the WebUI, navigate to: *Local Configuration > Security*.

Appliance Security Mode	Secure - (Default) ▼	?
HTTPS Port for Web User Interface	9443	?
Web Interface SSL Certificate	Default Self Signed Certificate ▼	?
Ciphers to use	ECDHE-ECDSA-AES256-GCM	?

Update

2. Select the required *Appliance Security Mode*.
3. If **Custom** is selected, configure the additional options to suit your requirements.
4. Specify the *HTTPS port for the Web User Interface* - the default is **9443**.



Note

This port must be the same as the port specified for the WebUI's HTTPS Service Socket Address. For more details, please refer to [Service Socket Addresses](#).

5. Select the required *Web Interface SSL Certificate* for the WebUI. If no certificates have been uploaded, the appliance's default self-signed certificate is used.



Note

Certificates can be created/uploaded using the WebUI menu option: *Cluster Configuration > SSL Certificate*.

6. Specify the required *ciphers to use* - the default is:

ECDHE-ECDSA-AES256-GCM-SHA384 : ECDHE-ECDSA-AES128-GCM-SHA256 : DHE-RSA-AES256-GCM-SHA384 : DHE-RSA-AES128-GCM-SHA256 : ECDHE-RSA-AES256-GCM-SHA384 : ECDHE-RSA-AES128-GCM-SHA256

7. Click **Update**.

Users & Passwords

Linux root Account

One of the great advantages of the Loadbalancer.org appliance is that full root access can be enabled. This unlocks the full capabilities of the underlying Linux OS. Other vendors tend to lock this down and only provide limited access to certain features and tools.



The "root" account is disabled by default. The account is enabled when a password is set using the [Network Setup Wizard](#). Once set in this way, it can be changed at the console or via an SSH session using the following command:

```
# passwd
```

Note

As mentioned in the section above, "root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI option: **Local Configuration > Security**. You'll need to Set **Appliance Security Mode** to **Custom**, enable the required option(s) and click **Update**.

Note

For the AWS and Azure cloud products it's not possible to directly login as "root". If root access is required, once you've logged into the console/SSH session using the credentials configured during instance deployment, run the following command:

```
$ sudo su
```

WebUI User Accounts

The appliance has 4 default WebUI user accounts:

Username	Default Group	Description
configuser	config	appliance administration account
loadbalancer	config (see note 1)	appliance administration account
reportuser	report	viewing the appliance configuration, reports & logs
maintuser	maint	same as reportuser plus can also take servers on/off line & create the support download archive file

Notes

1. It's not possible to change the group for the "loadbalancer" account.

All WebUI accounts are disabled by default. The "loadbalancer" account is enabled when a password is set using the [Network Setup Wizard](#). The "configuser", "reportuser" and "maintuser" accounts are enabled when a password is set as described below.

The permissions that apply to each group are as follows:

Menu / Permissions								
Group	System Overview	Local configuration	Cluster Configuration	Maintenance	View Configuration	Reports	Logs	Support
report	View	None	None	None	View	Full	View	View
maint	Full	None	None	None	View	Full	View	Full



Menu / Permissions								
config	Full	Full	Full	Full	View	Full	View	Full

Configuring Passwords

To set a user's password:

1. Using the WebUI, navigate to: *Maintenance > Passwords*.

Passwords

configuser	Modify	Delete
loadbalancer	Modify	
maintuser	Modify	Delete
reportuser	Modify	Delete

2. Click **Modify** next to the relevant user.

Username	<input type="text" value="loadbalancer"/>
Password *	<input type="password"/>
Re-enter Password *	<input type="password"/>

3. Enter the required password.



Note

Permitted password characters are: **a-z A-Z 0-9 [] # ~ _ * ! = -**

4. Click **Edit User**.

Adding New Users

To add a new user:

1. Using the WebUI, navigate to: *Maintenance > Passwords*.



2. Scroll down to the *Add New User* section.

Add New User

Username

LDAP/ADAuth User

☐

Password *

Re-enter Password *

Group

report ▾

Add New User

3. Enter the required *Username*.
4. If the user will be authenticated by an external LDAP/ADAuth or RADIUS system, enable (check) the *LDAP/ADAuth User* checkbox.



Note

For more information on external authentication, please refer to [External Authentication](#) below.

5. For locally authenticated users, enter the required *Password*.



Note

Permitted password characters are: a-z A-Z 0-9 [] # ~ _ * ! = -

6. Select the required *Group* for the new user.
7. Click **Add New User**.

Resetting forgotten Passwords

It's possible to reset passwords via the command line if required. To do this login as "root" at the console or via an SSH session and run the following command:

```
htpasswd -b /etc/loadbalancer.org/passwords loadbalancer <new password>
```



Note

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, configure the required option(s) and click **Update**.

External Authentication

The appliance supports the following external authentication methods:



- LDAP
- LDAPS
- Active Directory
- Radius

Once a user is configured to use external authentication, they must use their credentials for that system to access the appliance. It's important to remember that the username specified in the **Add New User** screen must be the exact same username used in the external authentication system.

To demonstrate this feature, an LDAP example is presented below. The steps required to configure other external authentication systems are similar.

Configuring External LDAP Authentication

To configure external authentication, login as "root" at the console or via SSH and run the following command:

```
lbauthconfig
```

Note

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, configure the required option(s) and click **Update**.

Note

Do not run lbauthconfig from the WebUI.

The following will be displayed:

```
[root@lbmaster lbadmin]# lbauthconfig
#####
#   Loadbalancer.org External Authentication Config Script   #
#####

This script will setup either LDAP or RADIUS authentication for the Load Balancer WUI.
It should be noted that it does not disable the local user access for users such as
"loadbalancer", "maintenance" and "reports" (Or any other additional locally created users)
so it is recommended that once you have tested Auth works successfully that you then assign
long and complex passwords to these users and file them away.

It should also be noted that you will need to add users to the load balancer in order for them
to gain access, authentication is handled by the external authentication method but access is
still controlled by the load balancers default groups so local users identical to those stored
in the external auth provider are required.

Step 1.  Select authentication type.
1. Local Files (Reset to defaults)
2. LDAP
3. LDAPS
4. Active Directory
5. Radius
```



[1-5]:

- To configure LDAP, enter **2** and hit <ENTER>

Step 2. Enter the hostname/address and port of the authentication server.

Hostname/address: 192.168.112.1

Port [1-65535 (389)]:

- Specify the hostname or IP address of your LDAP server, e.g. **192.168.112.1** and hit <ENTER>.
- Leave the Port blank to use the default value (389) or specify a different port and hit <ENTER>.

Step 3. Enter the LDAP Base Search string.

In the case of AD at a minimum this is your domain name so for "DOMAIN.LOCAL" you would use DC=DOMAIN,DC=LOCAL.

Search Base: DC=lbtestdom,DC=com

- Specify the LDAP search base string for your domain, e.g. **DC=lbtestdom,DC=com** and hit <ENTER>.

Step 4. Enter the LDAP attribute to authenticate against.

In the case of AD this will be "userPrincipalName" or "samAccountName" while OpenLDAP will typically use "uid".

Attrib [uid]: UserPrincipalName

- Specify the LDAP attribute to authenticate against, e.g. **UserPrincipalName** and hit <ENTER>.

Step 5. Enter the credentials for a user who can browse the directory.

Username: lbuser@lbtestdom.com

Password:

- Specify the credentials for a user who can browse the directory, e.g. **lbuser@lbtestdom.com** and hit <ENTER>.

 **Note**

This account must be able to list the current users. It should not be deleted since it is also used when additional users are added. If the password is changed, steps 1 - 6 must be repeated.

Step 6. Please add your first user.

This user will be added as a "config" user with full access to the WUI.

Please add additional users via the WUI after this setup process.

Username: tom@lbtestdom.com

Password:

- Specify the username of the first LDAP authenticated user, e.g. **tom@lbtestdom.com** and hit <ENTER>.

- Specify the password for the user and hit <ENTER>.

```
Test authentication succeeded.
```

```
Creating backup of /etc/httpd/webui.conf.d/10-lbauthconfig-settings.conf at
/etc/loadbalancer.org/bkup/10-lbauthconfig-settings.conf-2023-03-31T12:58:42.698035
Creating backup of /etc/loadbalancer.org/groups at /etc/loadbalancer.org/bkup/groups-2023-03-
31T12:58:42.698035
Writing new /etc/httpd/webui.conf.d/10-lbauthconfig-settings.conf File.
Writing new /etc/loadbalancer.org/groups File.
```

```
Finished.
```

The new user will be added to the appliance and can be viewed using the WebUI menu option: **Maintenance > Passwords**. When the user logs in they must use their LDAP credentials.

Adding Additional Users



Note

Configuring external authentication using groups is currently not supported.

Once the first user has been added, additional users can be added using the **Add New User** screen which is accessible via the WebUI option: **Maintenance > Passwords** as shown below:

Add New User

Username	<input type="text" value="tim@lbtestdom.com"/>
LDAP/ADAuth User	<input checked="" type="checkbox"/>
Group	<input type="text" value="report"/>

Add New User

1. Specify the username of the user to be added, e.g. **tim@lbtestdom.com**.
2. Enable (check) the **LDAP/ADAuth User** checkbox.
3. Select the required appliance security **Group**.
4. Click **Add New User**.

User **tim@lbtestdom.com** will now be able to login to the appliance with report user access rights using his LDAP credentials.

Firewall Configuration

The firewall can be configured using the WebUI script editor. This enables iptables rules and any other required commands to be easily configured. The editor allows you to directly edit /etc/rc.d/rc.firewall.



Note

Whilst the load balancer is capable of supporting complex firewall rules, we do not recommend using the load balancer as your main bastion host. We recommend that the load balancer is deployed behind your external firewall.

To edit the firewall script:

1. Using the WebUI, navigate to: **Maintenance > Firewall Script**.
2. The following screen will be displayed:

Firewall Script

```
1  #!/bin/sh
2  # $Id$
3
4  #
5
6  # User firewall script for Loadbalancer.org appliance.
7  #
8
9
10
11 # Please note:
12 #      Most configurations will not require any changes to be made to
13 #      this script.
14 #
15 #
16 #      Administrators will only need to modify this script if their
17 #      needs are not met by the lock-down wizard, auto-NAT, and
18 #      automatic firewall mark functions of the web interface.
19 #
20
21
22
23 ##### One-arm NAT Mode #####
24 # For one-arm NAT, ICMP re-directs will need to be disabled.
25 # (1 = on, 0 = off)
26 #echo "0" >/proc/sys/net/ipv4/conf/all/send_redirects
27 #echo "0" >/proc/sys/net/ipv4/conf/default/send_redirects
28
29
30
31
32 ##### Manual Firewall Marks #####
```

Update

3. Specify additional rules anywhere in the script above the last two lines:

```
echo "Firewall Activated"
exit 0;
```

4. Click **Update**.

Important

For a clustered pair, make sure you configure the firewall script in the same way on both appliances.

Firewall Lock-down Wizard

The firewall lock down wizard can be used to automatically configure the load balancer to allow access to the various admin ports from one specific IP address or subnet. The wizard automatically detects the IP address of the client running the WebUI and inserts this into the Admin IP field. The default mask is set to 255.255.255.0 which can be changed as required.



The firewall lockdown wizard uses two files:

- **rc.lockdownwizard** - this file contains the script that can be changed.
- **rc.lockdownwizard.conf** - this file contains a set of variable definitions and is written automatically when the **Update firewall lock down** button is clicked. The file depends on the rc.lockdownwizard script and the load balancer's current configuration. This file should not be changed manually.

When run, rc.lockdownwizard loads the settings from the definitions file rc.lockdownwizard.conf and uses them to generate the firewall rules. You can modify rc.lockdownwizard via ssh or from the WebUI using the **Modify the firewall lock down wizard script** button.

Note

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: **Local Configuration > Security**. You'll need to Set **Appliance Security Mode** to **Custom**, configure the required option(s) and click **Update**.

The default script does not depend on the configured Virtual Services or Real Servers, so the wizard does not need to be re-run when services are changed. However, it does depend on the IP addresses of the Primary and Secondary appliance, and the admin related ports used by the WebUI, heartbeat, and HAProxy. If those settings are changed, the firewall lockdown wizard will need to be re-run in order to reflect the changes. Re-running the firewall lockdown wizard will adapt the rc.lockdownwizard.conf definitions file automatically - any changes made to the script rc.lockdownwizard will remain when you re-run the firewall lockdown wizard.

To run the lock-down wizard:

Note

If you plan to configure an HA pair, either run the lockdown script on each appliance after the pair has been configured, or if it has already been run, temporarily disable the script (on both appliances) whilst performing the pairing process.


1. Using the WebUI, navigate to: **Maintenance > Firewall Lock Down Wizard**.
2. The following screen will be displayed:

Firewall Lock Down Wizard


WARNING: Once the lock-down wizard is enabled, administration access to the load balancer will only be allowed from the Administration Subnet specified below.

Enable lock down script

☒



Administration subnet



Update firewall lock down

Modify the firewall lock down wizard script

3. Enable (check) the **Enable lock down script** checkbox.

- Specify the management subnet or host address in the **Administration subnet** field.

Note

To specify additional management subnets/hosts, click the **Modify the firewall lock down wizard script** button and scroll down to the IPV4_NET_ADMIN parameter. Uncomment the line and specify additional subnets separated by spaces, e.g.

```
IPV4_NET_ADMIN=(192.168.4.0/24 192.168.6.0/24)
```

- Click **Update firewall lock down**.

To disable the lock-down script:

- To disable the lock-down script uncheck the **Enable lock down script checkbox** and click the **Update Firewall lock down** button.

Note

If you accidentally block your own access to the appliance you will need to clear the current firewall rules and try again. To completely clear the firewall rules use the following command at the console:

```
/etc/rc.d/rc.flush-iptables
```

Important

For a clustered pair you'll need to run/configure the firewall lockdown wizard/script in the same way on both appliances.

Conntrack Table Size

By default the connection tracking table size is set to 524288 which works well in most cases. For high traffic deployment using NAT mode, or when using connection tracking in the firewall script, this value may need to be increased. If the connection tracking table fills up, the following error will be reported in the log:

```
nf_conntrack: table full, dropping packet
```

To configure the table size:

- Using the WebUI, navigate to: **Local Configuration > Physical - Advanced Configuration**.
- Scroll down to the **Firewall** section.
- Set **Connection Tracking table size** to the required value.
- Click **Update**.

Appliance Security Lockdown Script

To ensure that the appliance is secure it's recommended that a number of steps should be carried out. These steps have been incorporated into a lockdown script which can be run at the console (recommended) or via an SSH session. When run on the Primary of a correctly configured clustered HA pair, both appliances will be



updated. The script locks down the following:

- the password for the "loadbalancer" WebUI account
- the password for the Linux "root" account
- from which subnet/host WebUI and SSH access is permitted

It also regenerates the SSH keys that are used to secure communicating between the Primary and Secondary appliance. To start the script, at the console or via an SSH terminal session run the following command:

```
lbsecure
```

Note

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: **Local Configuration > Security**. You'll need to Set **Appliance Security Mode** to **Custom**, configure the required option(s) and click **Update**.

The following illustrates how the script works for a single appliance:

Loadbalancer.org security lock-down

This script enhances the security of a single or high-availability pair of load balancers.

You will be asked to provide new passwords for the web interface and the console root account, plus an IP subnet that should be allowed remote access to the load balancer's web interface and ssh console.

Please enter a new password for the web interface 'loadbalancer' user. The password will not be displayed as you type.

New web interface password: *****

Confirm password:

Please enter a new password for the console 'root' user. The password will not be displayed as you type.

This password will also be used for the console 'setup' user.

New console password:

Confirm password:

Please enter an IP subnet that should be allowed remote access to the web interface and ssh console.

Note that any host outside of this subnet will immediately lose access to the load balancer. If you are running this script remotely, that includes the current console.

Administration subnet: 192.168.10.0/24

Working...

Setting web interface password...

Setting console root password on local machine...

Setting console 'setup' password on local machine...

Passwords set.

Setting up firewall...



```
Firewall enabled.  
Generating new SSH keys...  
SSH keys replaced.
```

```
Security enhancement complete.
```

Once the script has finished, the "**Security enhancement complete**" message is displayed as shown above.

Note

If `lbsecure` is run on the Primary of a correctly configured HA pair, the passwords, firewall rules and SSH keys will also be updated on the Secondary appliance.

You should run `lbsecure` **after** configuring the HA pair to ensure the correct HA related ports are configured in the firewall rules.

To reverse the action of `lbsecure`, the `lbinsecure` command can be used. For a clustered pair, run `lbinsecure` on both Primary and Secondary appliances to completely reverse the configuration applied by running `lbsecure`.

SSH Keys

This menu option enables SSH keys to be managed.

To view/manage SSH keys:

1. Using the WebUI, navigate to: *Local Configuration > SSH Keys*.

Host Keys ?

Create new key pair

Upload key pair

Type	Length (bits)	Date	
DSA	1024	2020-01-15 16:44	<div>Delete</div> <div>Download public key</div>
ECDSA	256	2023-01-20 17:38	<div>Delete</div> <div>Download public key</div>
ED25519	256	2023-01-20 17:38	<div>Delete</div> <div>Download public key</div>
RSA	2048	2020-01-15 16:44	<div>Delete</div> <div>Download public key</div>

User Keys ?

Create new key pair

Upload key pair

Username	Type	Length (bits)	Date	
root	RSA	2048	2020-01-15 16:44	<div>Delete</div> <div>Download public key</div>

Synchronise keys with peer

- The first tab (SSH Keys) enables the following keys to be viewed & managed:
 - **Host Keys** - the host identification key(s) of the local host
 - **User Keys** - the public key(s) of the user presented to remote hosts
- The second tab (SSH Authentication) enables the following keys to be viewed & managed:
 - **Host Keys (known_hosts)** - the key(s) of known hosts that have been previously connected to or have been preconfigured. For an HA pair the peer appliance's keys will be shown.
 - **User Keys (authorized_keys)** - the public key(s) of remote hosts that can log in as the specified user. For an HA pair the peer appliance's keys will be shown.

System Date & Time Configuration

Auto Configuration using NTP Servers



To configure NTP:

1. Using the WebUI, navigate to: *Local Configuration > System Date & Time*.

System Date & Time

Current system time

2023-01-31 15:42:33 UTC

System Timezone

UTC ▼

NTP Servers

Set Timezone & NTP

Date

2023 ▼ - Jan ▼ - 31 ▼

Time

15 : 42

Set Date & Time

2. Select the required *System Timezone*.
3. Specify the required NTP server(s) using the *NTP Servers* fields.
4. Click **Set Timezone & NTP**.

Manual Configuration

To manually set the date & time:

1. Set the data & time using the *Date & time* fields.
2. Click **Set Date & Time**.

Appliance Internet Access via Proxy

If required, a proxy server can be used to access the Internet.

To configure a proxy server:

1. Using the WebUI, navigate to: *Local Configuration > Physical Advanced Configuration*.
2. Scroll down to the *Network Proxy* section.



Network Proxy		
Proxy Server	<input type="text"/>	?
Port	<input type="text"/>	?
Username	<input type="text"/>	?
Password	<input type="password"/>	?

3. Enter the proxy's IP address in the **Proxy Server** field.
4. Enter the port in the **Port** field.
5. Enter a **Username** & **Password** if the proxy requires credentials.
6. Click **Update**.

SMTP Relay Configuration

The appliance can be configured to use an SMTP smart host to relay all email messages generated by the load balancer. If this field is not configured, the address will be auto-configured based on an MX lookup of the destination email address that's configured under *Cluster Configuration > Layer 4 - Advanced Configuration*.

To configure a smart host:

1. Using the WebUI, navigate to: *Local Configuration > Physical Advanced Configuration*.
2. Scroll down to the **SMTP Relay** section.

SMTP Relay	
Smart Host	<input type="text"/> ?

3. Enter an appropriate IP address or hostname in the **Smart Host** field.
4. Click **Update**.

Syslog Server Configuration

By default, all logs are written locally. Logs can be written to an external Syslog server if preferred, or to both locations.

To configure a syslog server:

1. Using the WebUI, navigate to: *Local Configuration > Physical Advanced Configuration*.
2. Scroll down to the **Logging** section.
3. Select either **Remote Syslog Server** or **Both**.

Logging		
Rate limit interval	<input type="text"/>	?
Rate limit Burst limit	<input type="text"/>	?
Log Destination	<input type="radio"/> Local Files <input checked="" type="radio"/> Remote syslog Server <input type="radio"/> Both	?
Remote syslog Server IP	<input type="text"/>	?
Remote syslog Server Port	<input type="text"/>	?
Remote syslog Server Protocol	UDP ▾	?
Remote syslog Server Template	<div style="border: 1px solid #ccc; height: 100px;"></div>	?

4. Specify the required *Rate limit interval*, the default is 5 seconds.
5. Specify the required *Rate limit Burst limit*, the default is 200 messages.
6. Specify the *Remote Syslog Server IP* - this can be an IP address or hostname.
7. Specify the *Remote Syslog Server Port*.
8. Specify the *Remote Syslog Server Protocol*.



Note

If the load balancer has been configured to keep detailed logs of multiple services, and your syslog server is heavily loaded, we recommend that UDP is used - this is the default.

9. Specify the required *Remote Syslog Server Template* in string format.
10. Click **Update**.

SNMP Configuration

The appliance supports SNMP v1, v2 and v3.

To configure SNMP:

1. Using the WebUI, navigate to: *Local Configuration > SNMP Configuration*.

Protocol Versions		
Enable SNMP v1 and v2	<input type="checkbox"/>	?
Enable SNMP v3	<input type="checkbox"/>	?
Details		
SNMP location	<input type="text" value="Unknown"/>	?
SNMP contact	<input type="text" value="IT Dept"/>	?
Authentication		
SNMP v1/v2 community string	<input type="text" value="public"/>	?
USM Username	<input type="text"/>	?
USM Authorization Algorithm	<input type="text" value="SHA"/>	?
USM Authorization Passphrase	<input type="text"/>	?
USM Privacy Algorithm	<input type="text" value="AES"/>	?
USM Privacy Passphrase	<input type="text"/>	?

Update

2. Enable the required SNMP version(s).
3. Enter the required **SNMP location** and **SNMP contact**.
4. For SNMP v1 & v2:
 - Enter the required **SNMP v1/v2 community string**.
5. For SNMP v3:
 - Specify the **USM Username**.
 - Select the required **USM Authorization Algorithm**.
 - Specify the **USM Authorization Passphrase**, it should be at least 8 characters.
 - Select the required **USM Privacy Algorithm**.
 - Specify **USM Privacy Passphrase**, it should be at least 8 characters.
6. Click **Update**.
7. Restart SNMPD using the **Restart SNMPD** button at the top of the screen.

Note

Valid characters for the **Community string**, **USM Username**, **USM Authorization Passphrase** and **USM Privacy Passphrase** fields are: **a-z A-Z 0-9 [] # ~ _ * ! = - \$ % ? { } @ : ; ^**

Note

For more information about the various OIDs and associated MIBs supported by the appliance, please refer to [SNMP Reporting](#).

Note

If you need to change the port, IP address or protocol that SNMP listens on, please refer to [Service Socket Addresses](#).

Running OS Level Commands

The appliance supports the ability to run OS level commands directly from the WebUI.

To run an OS level command:

1. Using the WebUI, navigate to: *Local Configuration > Execute Shell Command*.

Execute shell command

Execute shell command

2. Enter the required command in the field provided.
3. Click **Execute shell command**.

Note

Commands that run continuously when executed should be run with a specific count to ensure that they will terminate gracefully and the results can be displayed in the WebUI.

For example with ping use:

```
ping -c 4 192.168.100.254
```

Note

"Execute Shell Command" is disabled by default. This can be enabled using the WebUI menu option: *Local Configuration > Security*. Set *Appliance Security Mode* to **Custom** then click **Update**.

Portal Management & Appliance Adoption

The Loadbalancer.org ADC Portal is an ultra-secure, cloud-based ADC management platform that enables ADCs from Loadbalancer.org and multiple other vendors including F5, Citrix Netscaler and Progress Kemp to be monitored and managed. This enables centrally controlled backups, software updates, real-time security alerts (CVEs) and secure remote access for all connected ADCs. To add an appliance to the Portal, follow the steps below.

Step 1 - Configure Portal Connection Details

1. Using the WebUI, navigate to: *Local Configuration > Portal Management*.



Portal Management

Connection Details		[Advanced +]
Gateway Enabled	<input checked="" type="checkbox"/>	?
Shuttle Enabled	<input checked="" type="checkbox"/>	?
		Update

Adopt Appliance		
Adoption Submitted	no	
Portal Email	<input type="text" value="email@domain.com"/>	?
Portal Password	<input type="password" value="....."/>	?
		Begin Adoption

- To enable communication with the Portal, enable (check) the *Gateway Enabled* and *Shuttle Enabled* checkboxes.
- To view or change the hostname/IP address and port of the Portal click **[Advanced]**.



Note

In most cases the default values for *Hostname* and *Port* do not need to be changed.

- Click **Update**.
- To apply the new settings, the Gateway and Shuttle services must be restarted. This can be done using the buttons in the "Commit changes" box at the top of the screen.

Step 2 - Start the Appliance Adoption Process

- Enter the *Portal Email* & *Portal Password* for the account and organisation you'd like the appliance to be associated with.
- Click **Begin Adoption**.
- "Adoption Initiated" will be displayed in the blue information box.

Step 3 - Adopt the Shuttle

- Click **LOADBALANCER | PORTAL** in the Portal's main menu bar to view the Dashboard.
- in the *ADCs* panel:
 - If there are currently no ADCs, click **Connect an ADC**.
 - if ADCs have already been added, click **View my ADCs**.
- Using the menu on the left, select *Shuttle Management*.
- If the details have been entered correctly, the Shuttle for the appliance added will appear in the list as shown below.



ADCs

Overview

List

Shuttle Management

Shuttle Management

Search...

Shuttle Name	Running status	Associated ADCs	Namespace
Spirit Voyage II			<div>Adopt X</div>

1 total

Download CSV

Add Shuttle

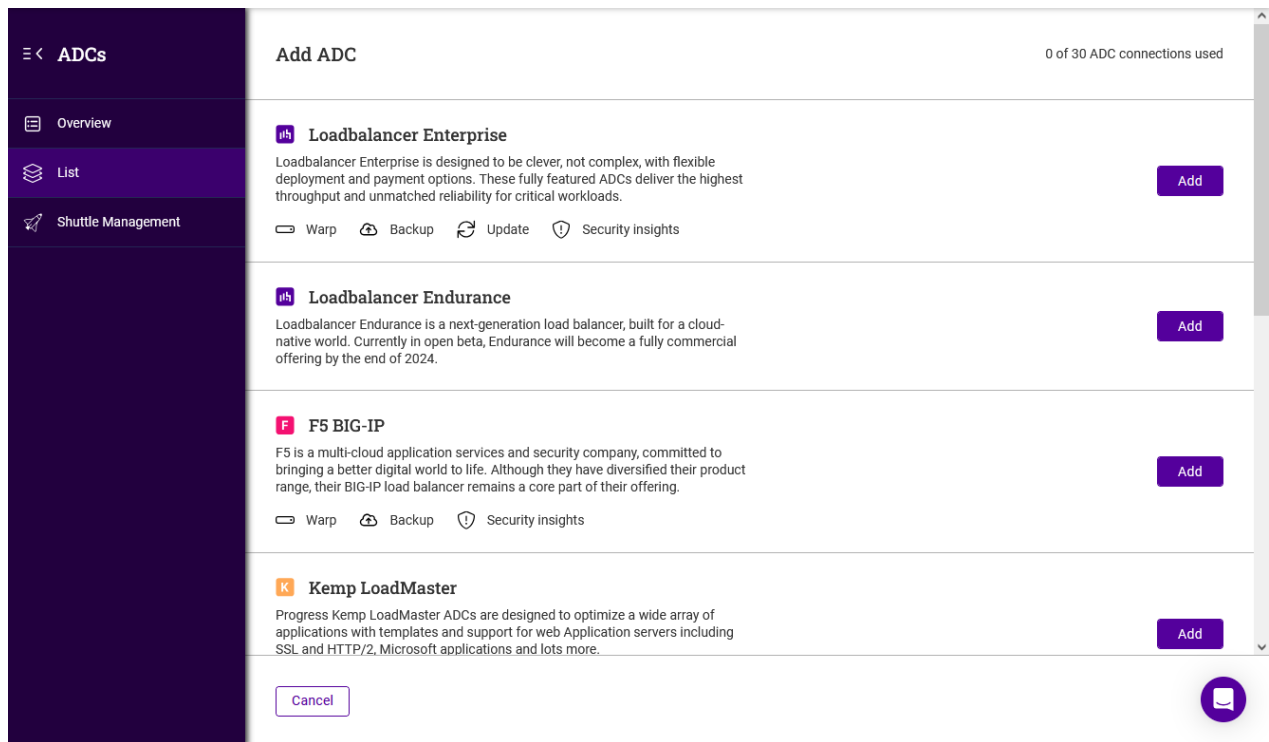
Note

A default name is allocated, this can be changed once the Shuttle is adopted. To edit the name, click the three dots menu to the right and select **Edit Shuttle**.

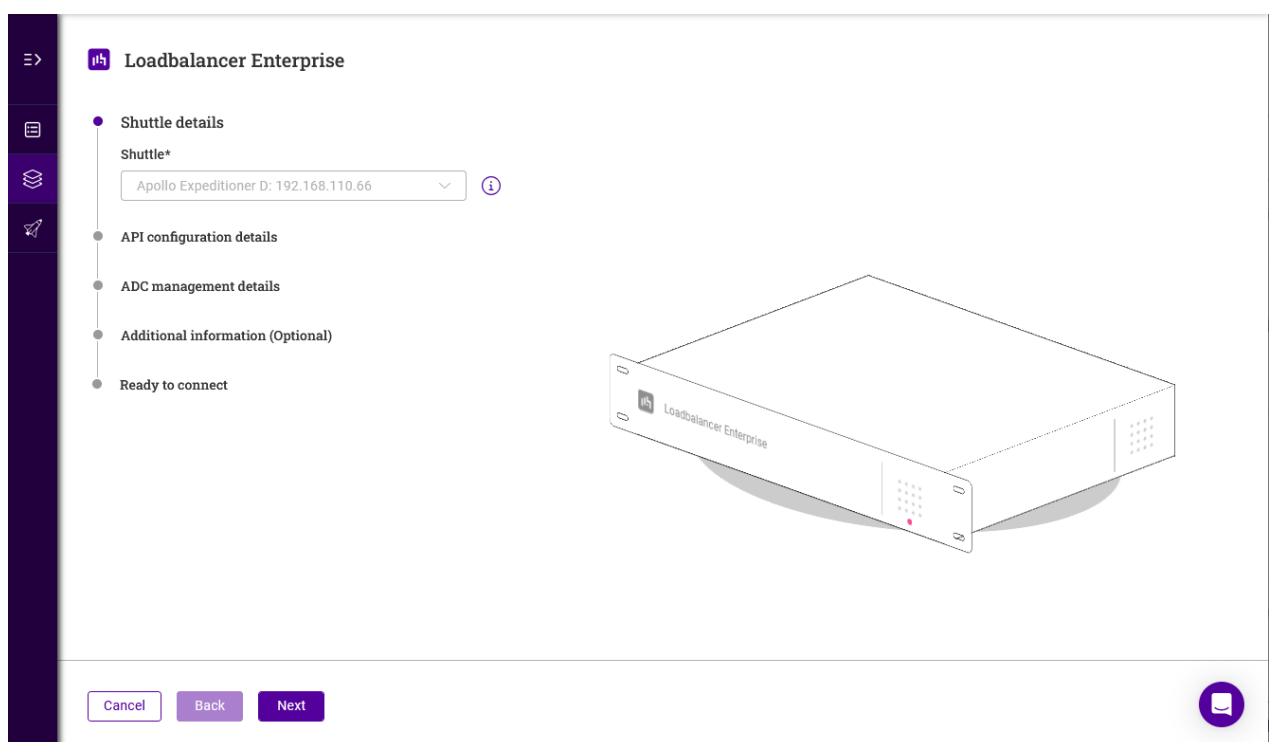
- Click the **Adopt** button for the new Shuttle to complete the Shuttle adoption process.

Step 4 - Add the ADC to the Portal

- Click **LOADBALANCER** | PORTAL in the Portal's main menu bar to view the Dashboard.
- In the **ADCs** panel:
 - If there are currently no ADCs, click **Connect an ADC**.
 - In the menu to the left, select **List**.
 - if ADCs have already been added, click **View my ADCs**.
- Click the **Add ADC** button.

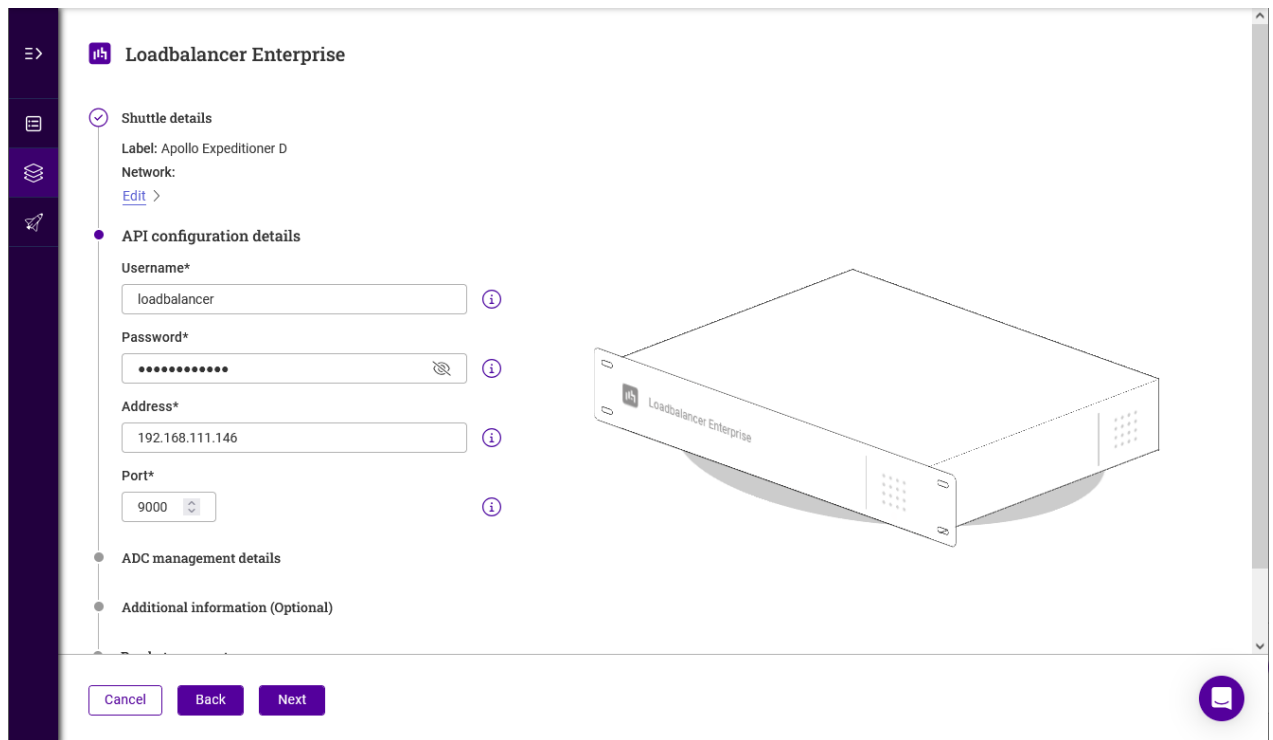


4. Click the **Add** button for the Loadbalancer Enterprise.



5. Using the **Shuttle** dropdown, select the shuttle that corresponds to the appliance being added. If there is only one shuttle available, it will be greyed out and selected automatically as shown above.

6. Click **Next**.



Loadbalancer Enterprise

Shuttle details

Label: Apollo Expeditionier D
Network:
[Edit >](#)

API configuration details

Username*
 ⓘ

Password*
 ⓘ

Address*
 ⓘ

Port*
 ⓘ

ADC management details

Additional information (Optional)

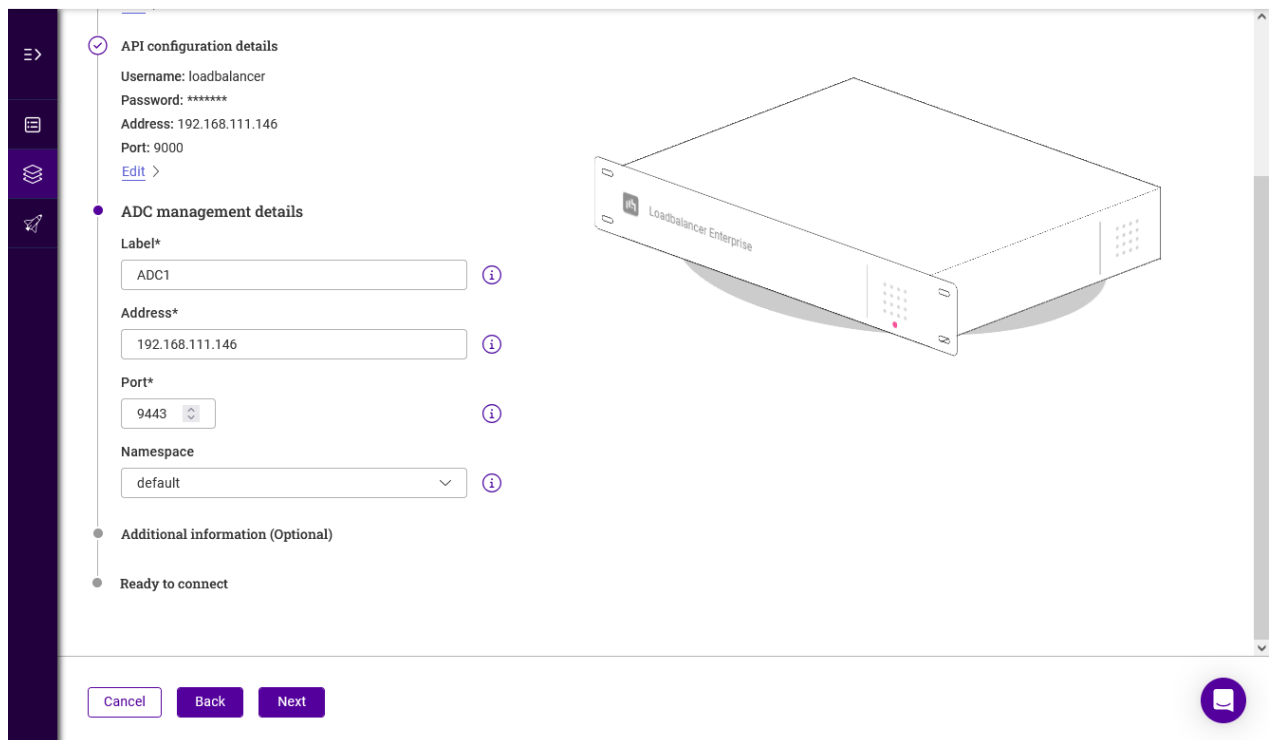
[Cancel](#) [Back](#) [Next](#)

7. Specify the "loadbalancer" **Username** and enter the associated **Password**.

8. Enter the **IP address** of the appliance being added.

9. Leave the **Port** set to the default value (**9000**).

10. Click **Next**.



API configuration details

Username: loadbalancer
Password: *****
Address: 192.168.111.146
Port: 9000
[Edit >](#)

ADC management details

Label*
 ⓘ

Address*
 ⓘ

Port*
 ⓘ

Namespace
 ⓘ

Additional information (Optional)

Ready to connect

[Cancel](#) [Back](#) [Next](#)

11. Enter an appropriate **Label** (name) for the appliance.

12. Ensure that the **IP Address** is correct.

13. Leave the **Port** set to the default value (**9443**).
14. Select the required **Namespace**.
15. Click **Next**.
16. Enter any required **Notes** and **Tags** to describe the appliance and click **Next**.



Note

To create a tag, enter the required name and hit <ENTER>. The tag will appear colored blue under the **Tags** field. Repeat to specify multiple tags (up to 30).

17. Verify all settings, these can be changed if needed using the relevant **Edit** option.
18. Click **Submit** - if the details have been specified correctly, the adopted appliance will appear in the list.



Note

The Portal connection method above uses a Shuttle and connection for each ADC. It's also possible to configure a single dedicated Shuttle and use this for all ADCs. For more information, please refer to the [Portal Quickstart Guide](#).

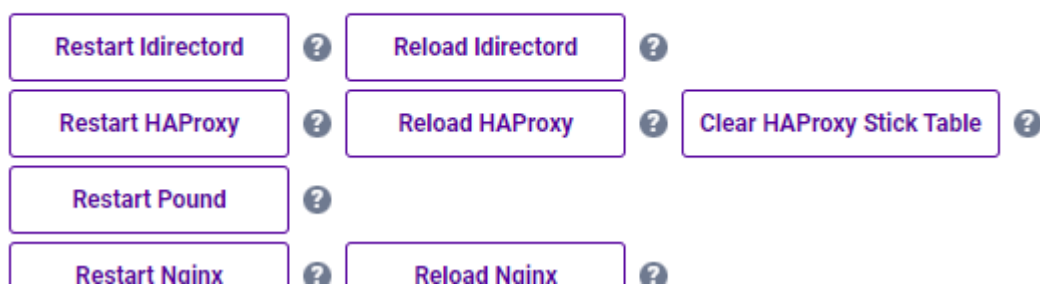
Restarting & Reloading Services

The various services running on the appliance can be manually reloaded or restarted if required. This is normally only required when a new configuration is added or an existing configuration is modified.

To restart / reload services:

1. Using the WebUI, navigate to: **Maintenance > Restart Services**.

Restart Services



2. Click the relevant restart or reload button.
3. When prompted, click **OK** to confirm you'd like to proceed.

The following restart & reload options are available:

Restart Ldirectord

Restarting Ldirectord will result in a loss of layer 4 services during the restart.

Reload Ldirectord

Reloading Ldirectord may result in a loss of layer 4 services during the reload.



Restart HAProxy

Restarting HAProxy will result in a loss of layer 7 services during restart. It will cause any persistence tables to be dropped and all connections to be closed.

Reload HAProxy

Reloading HAProxy will start a new process with the new configuration if settings have been changed and leave the current process running. New connections will be passed onto this new process, the old process will maintain existing connections and eventually terminate when there are no more connections accessing it. If you are using stick tables for persistence the entries will be copied between processes.

Note

If you have long lasting TCP connections it can take some time for the old HAProxy process to terminate. This means that those connections will continue to use the old configuration. If this is an issue, you can use Restart HAProxy instead.

Clear HAProxy Stick Table

If you are using stick table persistence, this will clear the entries for all tables. Clients may be directed to a different server upon re-connection.

Restart Pound

Restarting Pound will result in a loss of related SSL termination services during the restart.

Restart Nginx

Restarts the fallback web server. There will be a brief window of service interruption.

Reload Nginx

Reloads the configuration of the running fallback web server.

Restart STunnel

Restarting STunnel will result in a loss of related SSL termination services during the restart.

Reload STunnel

Reloading STunnel may result in a loss of related SSL termination services during the reload.

Restart Heartbeat

Restarting heartbeat will cause a temporary loss of all layer 4, layer 7 and SSL services.

Reload Heartbeat

Reloading heartbeat may cause a temporary loss of all layer 4, layer 7 and SSL termination services.

Restart Firewall

All firewall rules will be removed, then reloaded from the current configuration. This may result in a temporary loss of service.

Restart Syslogd

Restart Syslogd to apply any configuration changes.

Restart Collectd

Restart the graphing data collection daemon Previously collected data will not be lost. Note that collectd will not

start if graphing of all services is disabled.

Restart SNMPD

Restart the SNMPD service on the local system.

Reload Apache

Reload Apache performs a graceful restart which causes the parent process to advise the children to exit after their current request (or immediately if they're not serving anything). The parent then reloads its configuration and log files. As each child dies off the parent replaces it with a child with the updated configuration, which begins serving new requests immediately.

Restart WAF

Restarting the WAF will drop all current connections and re-read the config.

Reload WAF

Reload the WAF and re-read the config.

Restart GSLB

Restart GSLB services to make live any changes to configuration. This will impact live services.

Reload GSLB

Reload GSLB services to make live any changes to configuration. This should not impact live services.

Restart Gateway

Restart the loadbalancer management gateway service.

Restart SSH

Restart the SSH service. This will cause a brief outage of the service.

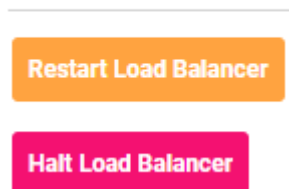
Appliance Restart & Shutdown

The appliance can be restarted or shutdown using the WebUI.

To restart or shutdown the appliance:

1. Using the WebUI, navigate to: *Maintenance > System Control*.

System Control



2. Select the required option.

Restoring Manufacturer's Settings



The load balancer can be reset to factory default settings in two ways. In both cases this will remove all user defined settings and configurations from the appliance and the IP address for **eth0** will be reset to 192.168.2.21/24.

Using the WebUI

To restore factory defaults:

1. Using the WebUI, navigate to: *Maintenance > Backup & Restore > Restore Tab*.
2. Click **Restore Manufacturer's Defaults**.
3. Once restored, restart the appliance to complete the process.

Using the Console / SSH Session

Run the following command:

```
lbrestore
```

Once restored, restart the appliance to complete the process.

Note

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, configure the required option(s) and click **Update**.

Chapter 6 - Configuring Load Balanced Services

Introduction

As discussed in [Layer 4 vs Layer 7](#), a fundamental choice when setting up load balanced services is whether to configure the services at layer 4 or at Layer 7.

Layer 4 Services

The Basics

Layer 4 services are based on LVS (Linux Virtual Server). LVS implements transport layer load balancing inside the Linux kernel. It is used to direct requests for TCP/UDP based services to the Real Servers and makes services on the Real Servers appear as a Virtual Service on a single IP address.

With the exception of Layer 4 SNAT mode, Layer 4 services are transparent by default, i.e. the source IP address is maintained through the load balancer to the Real Servers.

Layer 4 persistence is based on source IP and is enabled by default. The time out value is in seconds and each time the client makes a connection the timer is reset, so a 5 minute persistence setting could last for hours if the client is active and regularly refreshes their connection.

When a VIP is added, the load balancer automatically adds a corresponding floating IP address which is activated instantly. You can use the WebUI option: **View Configuration > Network Configuration** to ensure that the floating IP address is active. It will be listed as a secondary address/alias.

Multiple ports can be specified, for example 80 & 443. In this case persistence is useful to ensure that clients hit the same backend server for both HTTP & HTTPS traffic and also to prevent the client having to renegotiate the SSL connection.

Note

It's not possible to configure a VIP on the same IP address as any of the network interfaces. This ensures services can "float" (move) between Primary and Secondary appliances when using an HA Pair.

Creating Layer 4 Virtual Services

Virtual services (VIPs) can be configured either by creating a new VIP and configuring all required settings, or by using the duplicate VIP feature.

Each Virtual Service can have an unlimited number of Real Servers. Typically you'll need one Virtual Service for each distinct cluster (group of load balanced servers). For example, you could create a VIP for a web cluster, another for an FTP cluster and a third for a SIP cluster.

Configuring a New Layer 4 VIP

To add a new layer 4 VIP:

1. Using the WebUI, navigate to: **Cluster Configuration > Layer 4 - Virtual Services**.
2. Click **Add a new Virtual Service**.

Layer 4 - Add a new Virtual Service

Virtual Service		
Label	<input type="text" value="VIP Name"/>	?
IP Address	<input type="text" value="10.0.0.20"/>	?
Ports	<input type="text" value="80"/>	?
Protocol		
Protocol	<input type="text" value="TCP"/>	?
Forwarding		
Forwarding Method	<input type="text" value="Direct Routing"/>	?
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

- Enter an appropriate **Label** (name) for the new Virtual Service.
- Enter the required IP address in the **IP address** field.



Note

For the Virtual Service to listen on all IP addresses configured on the appliance (including the management address) specify **0.0.0.0** in the **IP address** field.

- Enter the required port(s) in the **Ports** field. Individual port numbers should be separated by commas, and ranges may be specified using a dash. To specify all ports, use a single asterisk. For example: 81, 443 - 447, 103, 104, 105. To exclude ports, use a single exclamation mark. Individual ports or ranges can be excluded. For example: *!80 or 400-500!450-460 or *(21-23,443,8080).



Note

The appliance uses a number of ports for various system functions. By default, most are bound to all appliance IPs and therefore cannot be used for Virtual Services unless reconfigured. For details, please refer to [Ports Used by the Appliance](#).

- Select the required **Protocol**:
 - **TCP** - Transmission Control Protocol is the default and most common option
 - **UDP** - User Datagram Protocol - used for DNS, SIP, etc.
 - **TCP/UDP** - enable both TCP and UDP on the port(s) specified
 - **One Packet Scheduling** - used for UDP SIP connections
 - **Firewall Marks** - For use when traffic has been tagged in the firewall script using the MARK target
- Select the required **Forwarding Method**:
 - **Direct Routing (DR)** - This is the default mode for new Layer 4 VIPs. To use this mode, the "ARP Problem" must be solved on each Real Server. For more information on DR mode, please refer to [Layer 4 DR Mode](#).



- **NAT** - With this mode, the Real Server's default gateway must be changed to be the load balancer. Because the load balancer also handles the return traffic, NAT mode is slower than DR mode. For more information on NAT mode, please refer to [Layer 4 NAT Mode](#).
- **Tunneling** - This is for WAN links (Tunneling). Tunneling has somewhat limited use as it requires an IP tunnel between the load balancer and the Real Server as the VIP is the target address many routers will drop the packet assuming that it has been spoofed. However, it is useful for private networks with Real Servers on multiple subnets.
- **SNAT** - The mode requires no Real Server changes but is not as fast as DR mode. Also it's non transparent and therefore loses the client source IP information. You should not use the same RIP:PORT combination for layer 7 SNAT mode VIPs and layer 4 SNAT mode VIPs because the required firewall rules conflict. For more information on SNAT mode, please refer to [Layer 4 SNAT Mode](#).

8. Click **Update**.

9. To configure the Real Servers, please refer to [Creating Layer 4 Real Servers \(RIPs\)](#).

Duplicating an Existing Layer 4 VIP

If you have existing Virtual Services, these can be duplicated using the **Duplicate Service** feature.

Note




This option copies all Virtual Service settings and the associated Real Servers. Once duplicated, you'll need to change either the IP address or port for the new VIP so that it does not clash with the original.


To duplicate an existing layer 4 VIP:



1. Click **Modify** next to the VIP you'd like to duplicate.
2. Click **Duplicate Service**.
3. Click **OK** at the prompt to confirm you'd like to proceed.
4. The VIP will be duplicated with a new label, all other settings will be identical.
5. Change either the **IP Address** or **Port** to ensure it does not clash with the source VIP.
6. Modify any other settings to suit your requirements.
7. Click **Update**.

Modifying a Layer 4 VIP

When a Virtual Service is first created, only certain settings can be configured. All other settings are set at default values to simplify initial configuration. These settings can be changed after the Virtual Service has been created by clicking **Modify** next to the Virtual Service. the following table details the additional settings that can be changed:

Section	Setting	Description
Connection Distribution Method	<i>Balance Mode</i>	<p>Select the required method to distribute new connections. The options are:</p> <ul style="list-style-type: none"> • Weighted Least-Connection (default) - assign more jobs to servers with fewer jobs, relative to the Real Server's weight. • Weighted Round Robin - assign jobs to Real Servers proportionally to the Real Server's weight. Servers with higher weights receive new jobs first and get more jobs than servers with lower weights. Servers with equal weights get an equal distribution of new jobs. • Destination Hash - assign jobs to servers through looking up a statically assigned hash table by their destination IP addresses. This algorithm is designed for use with web proxies and is supported with Layer 4 DR mode Virtual Services only. <div>  Note <p>When using destination hash, the web proxy servers must be configured in transparent mode as the destination remains set as the page a user requested. If the web proxy servers are configured in explicit/routed mode the destination will become the VIP. If the VIP is configured in either NAT or SNAT mode, the destination will be altered when the traffic is DNAT'ed flowing through the load balancer.</p> </div>
Persistence	<i>Enable</i>	<p>Enable (default) or disable source IP persistence.</p> <p>Sticky or persistent connections are required for some protocols such as FTP and SIP. It is also kind to clients when using SSL and is sometimes required with HTTP if your web application cannot keep state between Real Servers.</p> <div>  Note <p>If <i>Protocol</i> for the Virtual Service is set to One Packet Scheduling, persistence will be based on SIP Call-ID.</p> </div> <div>  Note <p>If your Real Servers cannot keep session state persistence themselves, then you will obtain performance but not reliability benefits from a load balancer.</p> </div>
	<i>Timeout</i>	<p>How long do you want connections to be sticky? The persistence time is in seconds and is reset on every connection. By default this is set to 300s (5 mins).</p>

Section	Setting	Description
	<i>Granularity</i>	Specify the granularity with which clients are grouped for persistent Virtual Services. The source address of the request is masked with this netmask to direct all clients from a network to the same Real Server. The default is 255.255.255.255, that is, the persistence granularity is per client host. Less specific netmasks may be used to resolve problems with non-persistent cache clusters on the client side.
Health Checks	<i>Check Type</i>	<p>Specify the type of health check to be performed on the Real Servers. As the Check Type dropdown is changed, the related field list changes. The options are:</p> <ul style="list-style-type: none"> • Negotiate - Scan the page specified in <i>Request to Send</i>, and check the returned data for the <i>Response Expected</i> string. • Connect to port - Attempt to make a connection to the specified port. • Ping Server - Use a simple ICMP ping to perform health checks. • External script - Use a custom file for the health check. For more information, please refer to External Health Check Scripts • No checks, always off - All Real Servers are marked offline. • No checks, always on - All Real Servers are marked online. • 5 Connects, 1 Negotiate - Repeating pattern of 5 Connect checks followed by 1 Negotiate check. • 10 Connects, 1 Negotiate - Repeating pattern of 10 Connect checks followed by 1 Negotiate check. <div>  Note For full details of all layer 4 health check options, please refer to Health Checks for Layer 4 Services. </div>
	<i>Check Port</i>	If you want the check port to be different to the port specified for the VIP, set it here. For a multi-port VIP, by default the first port in the list will be used as the check port.
Feedback	<i>Feedback Method</i>	<p>The method the load balancer uses to measure to performance of the Real Servers. The options are:</p> <ul style="list-style-type: none"> • Agent - A simple telnet to port 3333 on the Real Server. • HTTP - A simple HTTP GET to port 3333 on the Real Server. • None - No feedback (default setting). <p>The load balancer expects a 0-99 integer response from the agent, usually relating to the CPU idle; i.e. a response of 92 would imply that the Real Server's CPU is 92% idle. The load balancer will then use the formula $(92/100 * \text{requested_weight})$ to find the new weight.</p>

Section	Setting	Description
Fallback Server	IP Address	The server to route to if all of the Real Servers fail their health check. By default the fallback server is set to be the local NGINX instance.
		<div>  Note </div> <div>For more information on configuring the fallback server, please refer to Fallback Server.</div>
		You can also configure the fallback server to be a "hot spare" if required. To do this, configure the primary server as a Real Server and set the secondary server as the fallback server.
	Port	The port of the fallback server. For DR mode and TUN mode with masq disabled, the port should not be specified so that it defaults to the same as the Virtual Service.
	MASQ Fallback	Masquerade fallback. When enabled, this enables the fallback server to be set as a Layer 7 Virtual Service. This is especially useful in WAN/DR site environments.
	Email Alert Destination Address	Destination email address for server health check notifications.
		<div>  Note </div> <div>For more information on configuring email alerts, please refer to Configuring Email Alerts for Virtual Services.</div>



Tip

If you require a custom gateway for a particular VIP, this can be achieved using Policy Based Routing. For more information, please refer to [Policy Based Routing \(PBR\)](#).

Creating Layer 4 Real Servers (RIPs)

An Unlimited number of Real Servers can be configured for each VIP.


To add a new layer 4 RIP:

1. Using the WebUI, navigate to: **Cluster Configuration > Layer 4 - Real Servers**.
2. Click **Add a new Real Server** next to the relevant Virtual Service.

Label	<input type="text" value="RIP Name"/>	?
Real Server IP Address	<input type="text" value="IPAddress"/>	?
Real Server Port	<input type="text"/>	?
Weight	<input type="text" value="100"/>	?
Minimum Connections	<input type="text" value="0"/>	?
Maximum Connections	<input type="text" value="0"/>	?

- Enter an appropriate **Label** (name) for the new Real Server.
- Enter the required IP address in the **Real Server IP Address** field.
- Enter the required port in the **Real Server Port** field.

 **Note** For a multiport VIP, this field should be left blank.

 **Note** This field only applies to NAT mode and SNAT mode. In DR mode and TUN mode, port redirection is not possible so traffic is passed through on the same port that it was received on by the VIP.

- Specify the required **Weight**, this is an integer specifying the capacity of a server relative to the others in the pool, valid values are 0 to 65535, the default is 100. The higher the value, the more connections the server will receive. If the weight is set to 0, the server will effectively be placed in drain mode.
- Specify the **Minimum Connections**, this is an integer specifying the lower connection threshold of a server. The valid values are 0 through to 65535. The default is 0, which means the lower connection threshold is not set. If set, the server will receive new connections when the number of its connections drops below its lower connection threshold. If not set but **Maximum Connections** is set, the server will receive new connections when the number of its connections drops below 3/4 of its upper connection threshold.
- Specify the **Maximum Connections**, this is an integer specifying the upper connection threshold of a server. The valid values of Maximum Connections are 0 through to 65535. The default is 0, which means the upper connection threshold is not set.
- Click **Update**.

DR Mode Considerations

The ARP Problem

DR mode works by changing the MAC address of the inbound packets to match the Real Server selected by the load balancing algorithm. To enable DR mode to operate:

- The load balanced application/service/daemon running on each Real Server must be able to accept traffic destined for the VIP address **and** the Real Server's own IP address (RIP). This is because in DR mode the



destination address of load balanced packets is the VIP address, whilst for other traffic such as health checks, administration traffic etc. it's the Real Server's own IP address (the RIP).

2. Each Real Server must be configured so that it does not respond to ARP requests for the VIP address - only the load balancer should do this.

Configuring the Real Servers in this way is known as "Solving the ARP Problem". The steps required depend on the particular operating system being used.

Detecting the ARP Problem

Attempt to connect to the VIP and then use **Reports > Layer 4 Current Connections** to check whether the connection state is SYN_RECV as shown below.

Layer 4 Current Connections

Check Status

IPVS connection entries					
pro	expire	state	source	virtual	destination
TCP	00:26	SYN_RECV	192.168.64.7:20415	192.168.111.232:80	192.168.110.240:80
TCP	00:26	SYN_RECV	192.168.64.7:20414	192.168.111.232:80	192.168.110.240:80
TCP	04:18	NONE	192.168.64.7:0	192.168.111.232:80	192.168.110.240:80

If this is the case, it's normally a good indication that the "ARP Problem" has not been solved.

Solving the ARP Problem for Linux

There are two different approaches on how to configure a Linux server for correct operation when DR mode load balancing is in use:

- Modifying the server's ARP behavior and adding the relevant VIP addresses to the loopback interface
- Using NAT to convince the server to accept and reply to packets addressed to the relevant VIP addresses

Four independent methods are described below along with instructions. Each method follows one of the two approaches above. The specific method chosen will depend on technical requirements, the Linux distribution in use, and personal preferences.

The first method involves setting kernel parameters to alter the server's ARP behavior and adding IP addresses to the loopback interface. This method should be universally applicable to any Linux server **making this the preferred method**.

If setting kernel parameters and adding IP addresses is not possible for some reason, the remaining three methods describe setting up a server for DR mode operation by using NAT via the **redirect** target/statement. The specific instructions depend on the packet filtering framework and tooling in use, which varies between Linux distributions. Methods are presented for iptables, nftables, and the **firewall-cmd** tool.

Method 1: ARP Behavior and Loopback Interface Changes



This is the preferred method as it should be applicable to any Linux server and doesn't require any additional packet filtering or NAT considerations.

Each real server needs the loopback interface to be configured with the virtual IP addresses (VIPs) of the relevant load balanced services. This is often just a single VIP address, but the logic described below can be extended to cover multiple VIPs on a server. Having the VIPs on the loopback interface allows the server to accept inbound load balanced packets that are addressed to a VIP.

The server **must not** respond to ARP requests for the VIP addresses. The server also **must not** use ARP to announce the fact that it owns the VIP addresses. This is necessary to prevent IP address conflicts, as **all** of the real servers **and** the load balancer will own the VIP addresses. Only the load balancer should announce ownership of the VIPs.

To configure the behavior described above, follow all of the steps below on each real server.

Step 1 of 4: Re-configuring ARP behavior

This step is only applicable if IPv4-based virtual services are in use.

Add the following lines to the file `/etc/sysctl.conf` (create this file if it does not already exist):

```
net.ipv4.conf.all.arp_ignore=1
net.ipv4.conf.eth0.arp_ignore=1
net.ipv4.conf.eth1.arp_ignore=1
net.ipv4.conf.all.arp_announce=2
net.ipv4.conf.eth0.arp_announce=2
net.ipv4.conf.eth1.arp_announce=2
```

Adjust the commands shown above to suit the server's network configuration, e.g. a different number of network interfaces or a different interface naming convention.

For reference, the effect of these kernel parameter changes on the server is as follows:

Note

- **arp_ignore=1**: This configures the server to only reply to an ARP request if the request's target IP address is local to the incoming interface. This can never be true for VIP addresses on the loopback interface, as the loopback interface can never be an incoming interface for ARP requests from other devices. Hence, ARP requests for VIP addresses are always ignored.
- **arp_announce=2**: This prevents the server from sending an ARP request out of an interface **A** where the ARP request's sender/source address is stated to be an IP address that is local to some other interface **B**. For example, this prevents the server from sending an ARP request **from** a VIP address (which is local to the loopback interface) out of **eth0**, which would announce that the server owns the VIP address.

Step 2 of 4: Re-configuring duplicate address detection (DAD) behavior

This step is only applicable if IPv6-based virtual services are in use.

Add the following lines to the file `/etc/sysctl.conf` (create this file if it does not already exist):



```
net.ipv6.conf.lo.dad_transmits=0
net.ipv6.conf.lo.accept_dad=0
```

For reference, the effect of these kernel parameter changes on the server is as follows:

Note

- **dad_transmits=0**: This prevents a given interface from sending out duplicate address detection probes in order to test the uniqueness of unicast IPv6 addresses. Any IPv6 VIP addresses will *not* be unique, so this mechanism is disabled.
- **accept_dad=0**: This prevents a given interface from accepting duplicate address detection messages. This prevents any IPv6 VIP addresses from being marked as duplicate addresses.

Step 3 of 4: Applying the new settings

To apply the new settings, either reboot the real server or execute the following command to immediately apply the changes:

```
/sbin/sysctl -p
```

Steps 1, 2, and 3 can be replaced by instead modifying the necessary kernel variables by writing directly to their corresponding files under `/proc/sys/`. Note that changes made in this way *will not persist across reboots*.

Execute the following commands (as root) to implement these temporary changes (adapting the number of interfaces and interface names as needed):

Note

```
echo 1 > /proc/sys/net/ipv4/conf/all/arp_ignore
echo 1 > /proc/sys/net/ipv4/conf/eth0/arp_ignore
echo 1 > /proc/sys/net/ipv4/conf/eth1/arp_ignore
echo 2 > /proc/sys/net/ipv4/conf/all/arp_announce
echo 2 > /proc/sys/net/ipv4/conf/eth0/arp_announce
echo 2 > /proc/sys/net/ipv4/conf/eth1/arp_announce
echo 0 > /proc/sys/net/ipv6/conf/lo/dad_transmits
echo 0 > /proc/sys/net/ipv6/conf/lo/accept_dad
```

Step 4 of 4: Adding the virtual IP addresses (VIPs) to the loopback interface

Each of the VIP addresses must be permanently added to the loopback interface. VIPs must be added with a network prefix of /32 for IPv4 addresses or /128 for IPv6 addresses. The IP addresses can be added using the usual configuration files and tools for modifying network interfaces, which vary between different Linux distributions.

As an alternative, the `ip` command can be used as a universal way to add IP addresses to any Linux server. Note that addresses added in this way *will not persist across reboots*. To make these addresses permanent, add the `ip` commands to an appropriate startup script such as `/etc/rc.local`.

Execute the following `ip` command for each IPv4 VIP:



```
ip addr add dev lo <IPv4-VIP>/32
```

Execute the following `ip` command for each IPv6 VIP:

```
ip addr add dev lo <IPv6-VIP>/128
```

To check that the VIPs have been successfully added, execute the command:

```
ip addr ls
```

To remove an IPv4 VIP from the loopback adapter, execute the command:

```
ip addr del dev lo <IPv4-VIP>/32
```

To remove an IPv6 VIP from the loopback adapter, execute the command:

```
ip addr del dev lo <IPv6-VIP>/128
```

Method 2: NAT "redirect" via iptables

iptables can be used on each real server to identify incoming packets that are addressed to a virtual IP address (VIP) and redirect those packets to the server itself. This is achieved using the **REDIRECT** target in iptables, which performs the necessary NAT to make this possible. This allows a real server to accept packets addressed to a VIP without the server owning the VIP.

Execute the following command to put the necessary iptables rule in place to redirect traffic for a single IPv4 VIP address. Note that iptables rules added in this way **will not persist across reboots**. To make such a rule permanent, either add the rule to an iptables firewall script, if one is provided with the Linux distribution in question, or add the command to an appropriate startup script such as `/etc/rc.local` on each real server.

```
iptables -t nat -A PREROUTING -d <IPv4-VIP> -j REDIRECT
```

The VIP address should be changed to match the virtual service in question, for example:

```
iptables -t nat -A PREROUTING -d 10.0.0.21 -j REDIRECT
```

The example above will redirect any incoming packets destined for 10.0.0.21 (the virtual service) locally, i.e. to the primary address of the incoming interface on the real server.

If a real server is responsible for serving **multiple** VIPs then additional iptables rules should be added to cover each VIP.

For an IPv6 VIP address, a command like the following should be used:



```
ip6tables -t nat -A PREROUTING -d <IPv6-VIP> -j REDIRECT
```

The VIP address should be changed to match the virtual service in question, for example:

```
ip6tables -t nat -A PREROUTING -d 2001:db8::10 -j REDIRECT
```

Note

Method 2 may not be appropriate when using IP-based virtual hosting on a web server. This is because an iptables **REDIRECT** rule will redirect incoming packets to the *primary address* of the incoming interface on the web server rather than any of the virtual hosts that are configured. Where this is an issue, use method 1 instead.

Method 3: NAT "redirect" via nftables

nftables is the modern Linux kernel packet filtering framework. It is supported on all major Linux distributions and has replaced iptables as the default framework on most major distributions.

nftables can be used on each real server to identify incoming packets that are addressed to a virtual IP address (VIP) and redirect those packets to the server itself. This is achieved using the **redirect** statement in nftables, which performs the necessary NAT to make this possible. This allows a real server to accept packets addressed to a VIP without the server owning the VIP.

Use a script like the following to put the necessary nftables structures in place to redirect traffic for both IPv4 and IPv6 VIP addresses. To make such a configuration permanent, either add the **inet nat** table to an nftables firewall script, if one is provided with the Linux distribution in question, or configure a script like the following to execute as a startup script on each real server.

```
#!/usr/sbin/nft -f

table inet nat {
    chain prerouting {
        comment "Allow server to accept packets destined for VIP addresses";
        type nat hook prerouting priority -100; policy accept;
        ip daddr <IPv4-VIP> redirect comment "Description"
        ip6 daddr <IPv6-VIP> redirect comment "Description"
    }
}
```

The VIP addresses and comments should be changed to match the virtual services in question, for example:

```
#!/usr/sbin/nft -f

table inet nat {
    chain prerouting {
        comment "Allow server to accept packets destined for VIP addresses";
        type nat hook prerouting priority -100; policy accept;
        ip daddr 10.0.0.21 redirect comment "VIP 1: HTTP"
        ip6 daddr 2001:db8::10 redirect comment "VIP 2: HTTPS"
    }
}
```

```
}
```

The example above will redirect any incoming packets destined for 10.0.0.21 or 2001:db8::10 (the virtual services) locally, i.e. to the primary address of the incoming interface (for each IP version) on the real server.

Note that **Linux kernels prior to 5.2** may not support performing NAT (which is required for the **redirect** statement) in an inet family table. In this scenario, use either an ip or an ip6 family table instead, or both if a mixture of IPv4 and IPv6 VIPs are in use on the same server. Also note that older kernels may not support the use of comments in chains.

Note that **Linux kernels prior to 4.18** require explicitly registering both prerouting and postrouting chains in order for the implicit NAT of the **redirect** statement to be correctly performed in both the inbound and outbound directions.

A legacy-friendly setup may look like the following:

```
#!/usr/sbin/nft -f

table ip nat {
    chain prerouting {
        type nat hook prerouting priority -100; policy accept;
        ip daddr 10.0.0.21 counter redirect comment "VIP 1: HTTP"
    }

    chain postrouting {
        type nat hook postrouting priority 100; policy accept;
    }
}

table ip6 nat {
    chain prerouting {
        type nat hook prerouting priority -100; policy accept;
        ip6 daddr 2001:db8::10 counter redirect comment "VIP 2: HTTPS"
    }

    chain postrouting {
        type nat hook postrouting priority 100; policy accept;
    }
}
```

Note

Method 3 may not be appropriate when using IP-based virtual hosting on a web server. This is because an nftables **redirect** statement will redirect incoming packets to the **primary address** of the incoming interface on the web server rather than any of the virtual hosts that are configured. Where this is an issue, use method 1 instead.

Method 4: NAT "redirect" via firewall-cmd

Some recent versions of Linux distributions make use of firewalld as a high-level firewall configuration framework. In this case, while it may actually be iptables performing the work at a lower level, it may be preferred to implement the iptables NAT solution described in [method 2](#) in firewalld, as opposed to directly manipulating iptables. This is achieved by using the **firewall-cmd** tool provided by firewalld and executing a command like

the following on each real server:

```
firewall-cmd --permanent --direct --add-rule ipv4 nat PREROUTING 0 -d <IPv4-VIP> -j REDIRECT
```

The VIP address should be changed to match the virtual service in question, for example:

```
firewall-cmd --permanent --direct --add-rule ipv4 nat PREROUTING 0 -d 10.0.0.50 -j REDIRECT
```

To apply the new configuration, reload the firewall rules like so:

```
firewall-cmd --reload
```

Configuration applied in this way will be permanent and will persist across reboots.

Note

Method 4 may not be appropriate when using IP-based virtual hosting on a web server. This is because an iptables **REDIRECT** rule will redirect incoming packets to the *primary address* of the incoming interface on the web server rather than any of the virtual hosts that are configured. Where this is an issue, use method 1 instead.

Solving the ARP Problem for Solaris

For Solaris, the loopback interface does not respond to ARP requests so you just need add the VIP address to the interface:

```
ifconfig lo0:1 plumb  
ifconfig lo0:1 <VIP> netmask 255.255.255.255 up
```

You'll need to add this to the startup scripts on all of your Real Servers.

For Solaris v11 and later, a new command is used:

```
ipadm create-addr -a <VIP>/32 lo0
```

The configuration survives a reboot so there's no need to add this command to a startup script, just run it on each Real Server.

Solving the ARP Problem for Mac OS X/BSD

OS X is BSDish, so you need to use BSDish syntax:

```
ifconfig lo0 alias <VIP> netmask 255.255.255.255 -arp up
```

You'll need to add this to the startup scripts on all of your Real Servers.



Note

Don't forget that the service on the Real Servers needs to listen on both the RIP address and VIP address as mentioned previously.

Note

Failure to correctly configure the Real Servers to handle the "ARP Problem" is the most common issue when using DR mode.

Solving the ARP Problem for Windows Servers

Windows Server 2012 & Later

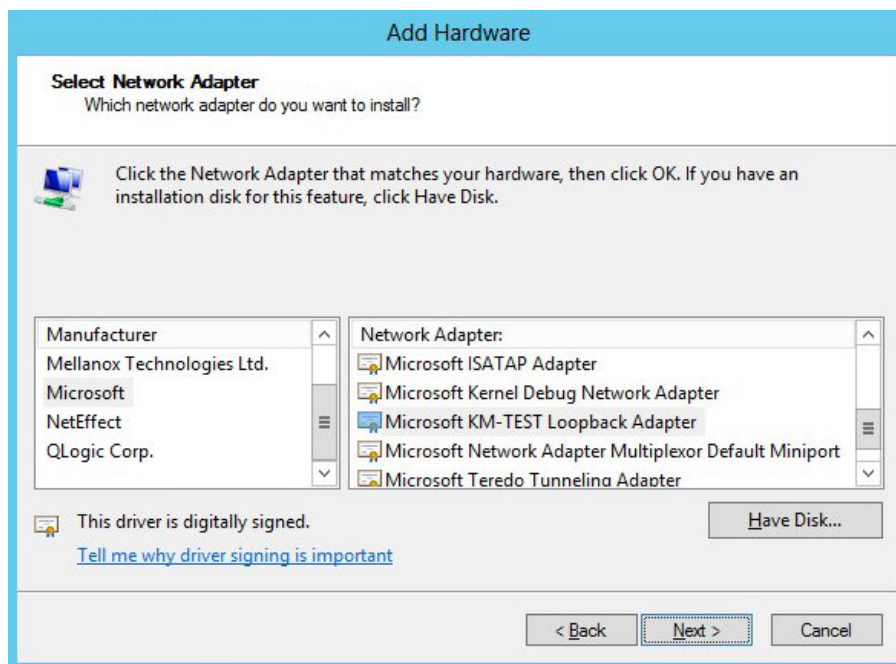
Windows Server 2012 and later support Direct Routing (DR) mode through the use of the Microsoft Loopback Adapter that must be installed and configured on each load balanced (Real) Server. The IP address configured on the Loopback Adapter must be the same as the Virtual Service (VIP) address. This enables the server to receive packets that have their destination set as the VIP address. If a Real Server is included in multiple DR mode VIPs, an IP address for each VIP must be added to the Loopback Adapter.

In addition, the strong/weak host behavior must be configured on each Real Server. The weak host model allows packets with any IP to be sent or received via an interface. The strong host model only allows packets with an IP belonging to the interface to be sent or received.

(!) Important The following 3 steps must be completed on **all** Real Servers associated with the VIP.

Step 1 of 3: Install the Microsoft Loopback Adapter

1. Click **Start**, then run **hdwwiz** to start the Hardware Installation Wizard.
2. Once the Wizard has started, click **Next**.
3. Select **Install the hardware that I manually select from a list (Advanced)**, click **Next**.
4. Select **Network adapters**, click **Next**.



5. Select **Microsoft & Microsoft KM-Test Loopback Adapter**, click **Next**.
6. Click **Next** to start the installation, when complete click **Finish**.

Step 2 of 3: Configure the Loopback Adapter

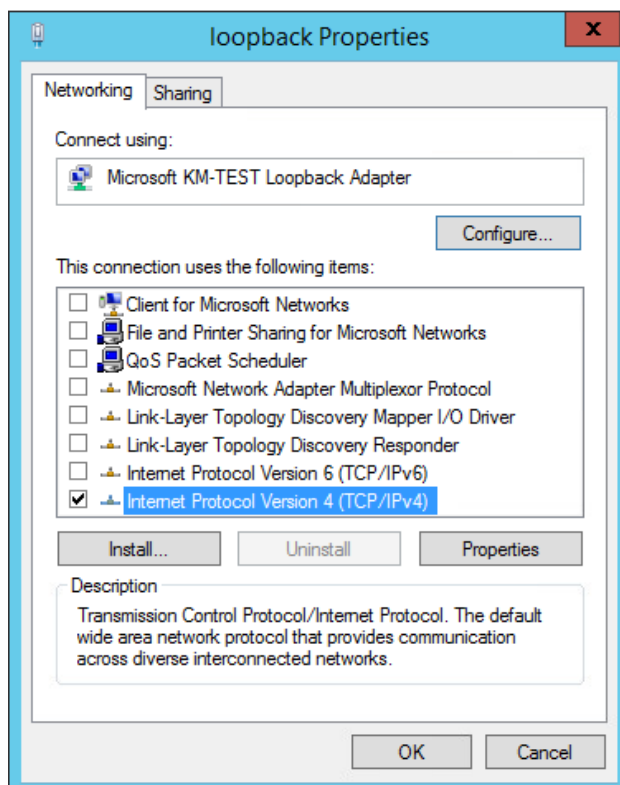
1. Open Control Panel and click **Network and Sharing Center**.
2. Click **Change adapter settings**.
3. Right-click the new Loopback Adapter and select **Properties**.

Note

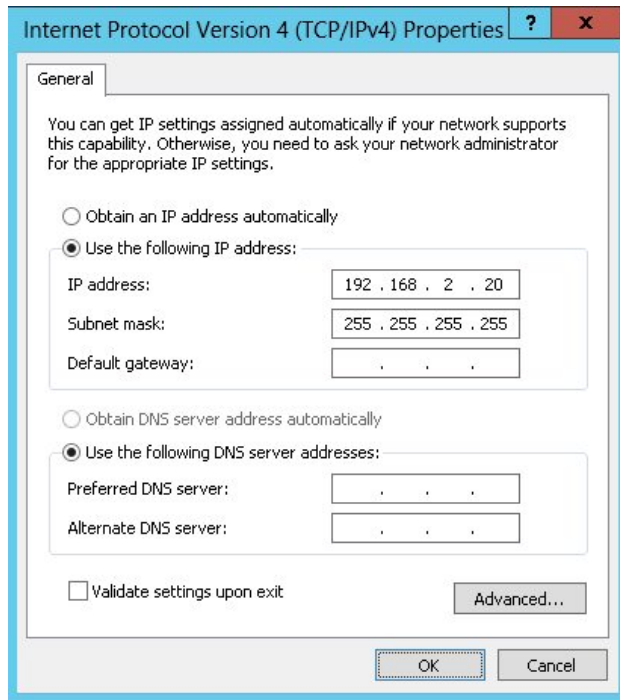
You can configure IPv4 or IPv6 addresses or both depending on your requirements.

IPv4 Addresses

1. Uncheck all items except **Internet Protocol Version 4 (TCP/IPv4)** as shown below:



2. Ensure that **Internet Protocol Version (TCP/IPv4)** is selected, click **Properties** and configure the IP address to be the same as the Virtual Service address (VIP) with a subnet mask of **255.255.255.255**, e.g. **192.168.2.20/255.255.255.255** as shown below:



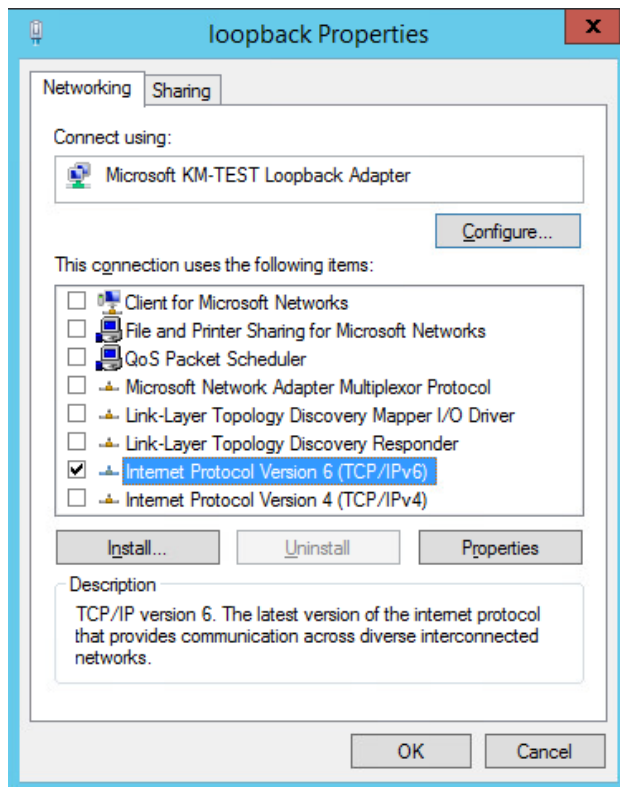
Note **192.168.2.20** is an example, make sure you specify the correct VIP address.

Note If a Real Server is included in multiple DR mode VIPs, an IP address for each VIP must be added to the Loopback Adapter.

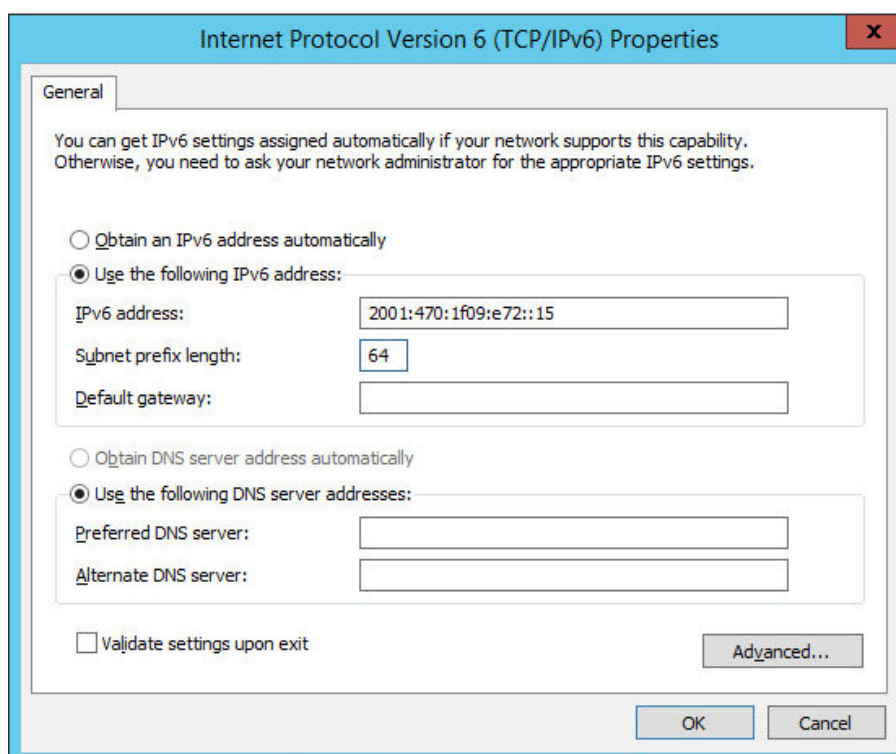
3. Click **OK** then click **Close** to save and apply the new settings.

IPv6 Addresses

1. Uncheck all items except **Internet Protocol Version 6 (TCP/IPv6)** as shown below:



2. Ensure that **Internet Protocol Version (TCP/IPv6)** is selected, click **Properties** and configure the IP address to be the same as the Virtual Service (VIP) and set the **Subnet Prefix Length** to be the same as your network setting, e.g. **2001:470:1f09:e72::15/64** as shown below:



Note **2001:470:1f09:e72::15/64** is an example, make sure you specify the correct VIP address.

Note If a Real Server is included in multiple DR mode VIPs, an IP address for each VIP must be

added to the Loopback Adapter.

3. Click **OK** then click **Close** to save and apply the new settings.

Step 3 of 3: Configure the strong/weak host behavior

The strong/weak host behavior can be configured using either of the following 2 methods:

- Option 1 - Using network shell (netsh) commands
- Option 2 - Using PowerShell cmdlets

The commands in this section assume that the LAN Adapter is named "**net**" and the Loopback Adapter is named "**loopback**" as shown in the example below:



(!) Important

Either adjust the commands to use the names allocated to your LAN and loopback adapters, or rename the adapters before running the commands. Names are case sensitive so make sure that the interface names used in the commands match the adapter names exactly.

Option 1 - Using Network Shell (netsh) Commands

To configure the correct strong/weak host behavior run the following commands:

For IPv4 addresses:

```
netsh interface ipv4 set interface "net" weakhostreceive=enabled
netsh interface ipv4 set interface "loopback" weakhostreceive=enabled
netsh interface ipv4 set interface "loopback" weakhostsend=enabled
```

For IPv6 addresses:

```
netsh interface ipv6 set interface "net" weakhostreceive=enabled
netsh interface ipv6 set interface "loopback" weakhostreceive=enabled
netsh interface ipv6 set interface "loopback" weakhostsend=enabled
netsh interface ipv6 set interface "loopback" dadtransmits=0
```

Option 2 - Using PowerShell Cmdlets

For IPv4 addresses:



```
Set-NetIpInterface -InterfaceAlias loopback -WeakHostReceive enabled -WeakHostSend enabled  
-DadTransmits 0 -AddressFamily IPv4
```

```
Set-NetIpInterface -InterfaceAlias net -WeakHostReceive enabled -AddressFamily IPv4
```

For IPv6 Addresses:

```
Set-NetIpInterface -InterfaceAlias loopback -WeakHostReceive enabled -WeakHostSend enabled  
-DadTransmits 0 -AddressFamily IPv6
```

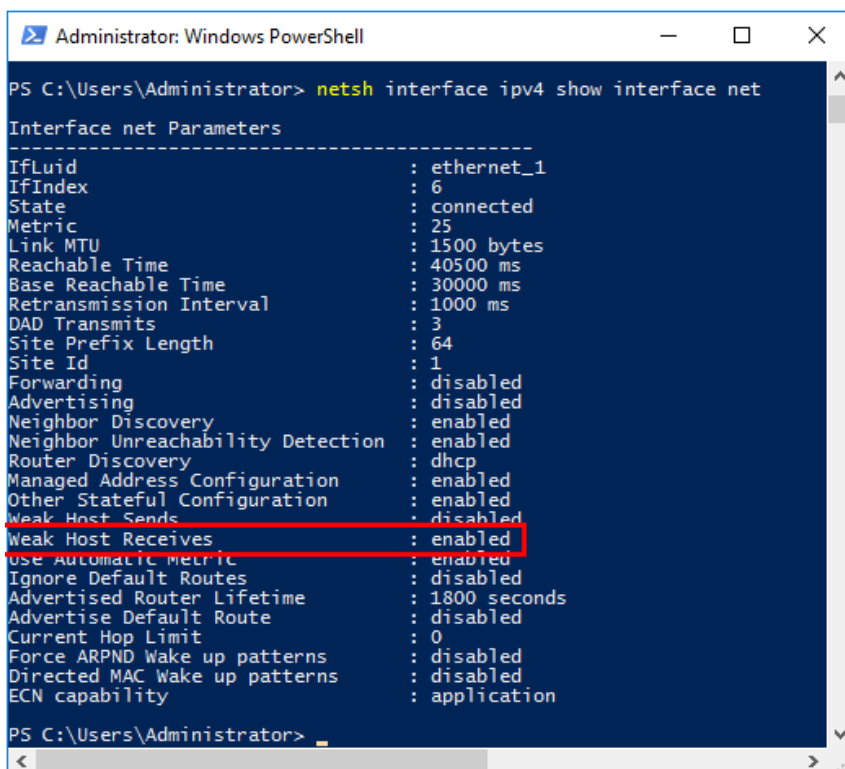
```
Set-NetIpInterface -InterfaceAlias net -WeakHostReceive enabled -AddressFamily IPv6
```

Verify the Strong/Weak Host Configuration

The following PowerShell Cmdlets can be used to verify the settings:

The "net" interface:

```
netsh interface ipv4 show interface net
```



```
Administrator: Windows PowerShell  
PS C:\Users\Administrator> netsh interface ipv4 show interface net  
Interface net Parameters  
-----  
IfLuid : ethernet_1  
IfIndex : 6  
State : connected  
Metric : 25  
Link MTU : 1500 bytes  
Reachable Time : 40500 ms  
Base Reachable Time : 30000 ms  
Retransmission Interval : 1000 ms  
DAD Transmits : 3  
Site Prefix Length : 64  
Site Id : 1  
Forwarding : disabled  
Advertising : disabled  
Neighbor Discovery : enabled  
Neighbor Unreachability Detection : enabled  
Router Discovery : dhcp  
Managed Address Configuration : enabled  
Other Stateful Configuration : enabled  
Weak Host Sends : disabled  
Weak Host Receives : enabled  
Use Automatic Metric : enabled  
Ignore Default Routes : disabled  
Advertised Router Lifetime : 1800 seconds  
Advertise Default Route : disabled  
Current Hop Limit : 0  
Force ARPND Wake up patterns : disabled  
Directed MAC Wake up patterns : disabled  
ECN capability : application  
PS C:\Users\Administrator>
```

The "loopback" interface:

```
netsh interface ipv4 show interface loopback
```

```
Administrator: Windows PowerShell

PS C:\Users\Administrator> netsh interface ipv4 show interface loopback

Interface loopback Parameters
-----
IfLuid           : ethernet_3
IfIndex          : 3
State            : connected
Metric           : 25
Link MTU         : 1500 bytes
Reachable Time   : 28000 ms
Base Reachable Time : 30000 ms
Retransmission Interval : 1000 ms
DAD Transmits    : 3
Site Prefix Length : 64
Site Id          : 1
Forwarding       : disabled
Advertising      : disabled
Neighbor Discovery : enabled
Neighbor Unreachability Detection : enabled
Router Discovery : dhcp
Managed Address Configuration : enabled
Other Stateful Configuration : enabled
Weak Host Sends  : enabled
Weak Host Receives : enabled
Use Automatic Metric : enabled
Ignore Default Routes : disabled
Advertised Router Lifetime : 1800 seconds
Advertise Default Route : disabled
Current Hop Limit : 0
Force ARPND Wake up patterns : disabled
Directed MAC Wake up patterns : disabled
ECN capability    : application

PS C:\Users\Administrator>
```

Note For IPv6, replace "ipv4" with "ipv6" in the above commands.

For Windows 2012 & later you can also use the following PowerShell Cmdlets to verify the settings:

To view both IPv4 and IPv6:

```
Get-NetIpInterface -InterfaceAlias loopback | FL
```

for IPv4 only:

```
Get-NetIpInterface -InterfaceAlias loopback -AddressFamily IPv4 | FL
```

for IPv6 only:

```
Get-NetIpInterface -InterfaceAlias loopback -AddressFamily IPv6 | FL
```

Note Failure to correctly configure the Real Servers to handle the "ARP Problem" is the most common issue when using DR mode.

Solving the ARP Problem - Possible Side Effect for Windows 2012 & Later

With DR Mode, the source IP address of return traffic from a Real Server will be the IP address assigned to the loopback adapter, which is the same as the VIP address that the client connected to. For traffic **initiated** by a Real Server, the source IP address should under normal circumstances be the Real Server's own IP address, i.e. the address assigned to the standard network adapter.

However, due to the way the network adapters are configured to solve the "ARP Problem", and the way that Windows selects the source IP address, it's possible under certain circumstances for the source IP address of traffic **initiated** by a Real Server to be the IP address configured on the loopback adapter rather than the Real Server's own IP address. Please refer to [this Microsoft article](#) for more information on how Windows selects the source IP address.

To prevent the IP address(es) assigned to the loopback adapter being used in this way, the **skipassource** flag must be set for each IP assigned to the loopback adapter.

To check the current **skipassource** setting for all IPs associated with the loopback adapter, use the following command:

```
Get-NetAdapter -name loopback | Get-NetIPAddress | select ipaddress, skipassource
```

- If your loopback adapter is not named "loopback" modify the command accordingly

Example output:

ipaddress	skipassource
192.168.111.180	false

This shows that for the 192.168.111.180 address, the skipassource flag is not set.

To set the skipassource flag for the 192.168.111.180 address, use the following command:

```
Set-NetIPAddress -IPAddress 192.168.111.180 -SkipAsSource $True
```

To clear the skipassource flag for the 192.168.111.180 address, use the following command:

```
Set-NetIPAddress -IPAddress 192.168.111.180 -SkipAsSource $False
```

To set the skipassource flag for **all** IP addresses associated with the loopback adapter, the following two PowerShell commands can be used:

```
[array]$IPs = Get-NetIPAddress -InterfaceAlias loopback
```

```
Set-NetIPAddress -IPAddress $IPs.IPAddress -InterfaceAlias loopback -SkipAsSource $true
```

- The first command gathers all IP addresses assigned to the loopback adapter
- The second command then sets the **SkipAsSource** flag for all IPs found
- If your loopback adapter is not named "loopback" modify the commands accordingly

Note

Make the sure that the same settings are configured on all Real Servers.

Other Windows Settings that May Cause Issues

Receive Segment Coalescing (RSC)

RSC is a stateless offload technology that helps reduce CPU utilization for network processing on the receive side by offloading tasks from the CPU to an RSC-capable network adapter. In rare cases it has been discovered that RSC can adversely effect performance when using DR mode. In these cases the performance issue was addressed by disabling RSC.

To check the current RSC settings for all RSC compatible interfaces, use the following command:

```
Get-NetAdapterRsc
```

Example output:

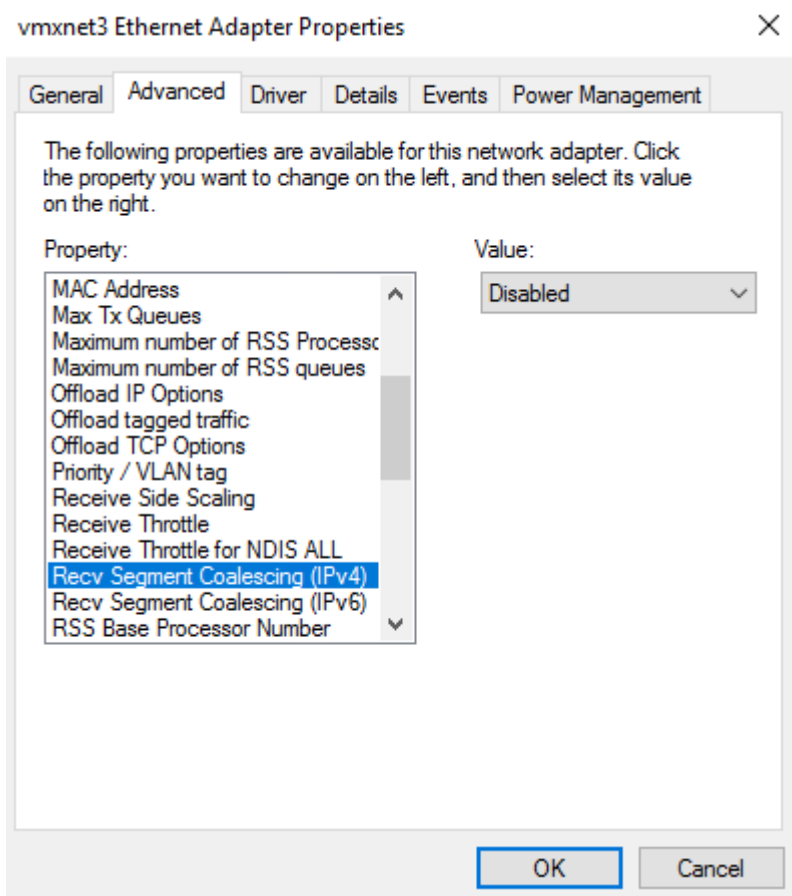
Name	IPv4Enabled	IPv6Enabled	IPv4Operational State	IPv6Operational State	IPv4FailureReason	IPv6FailureReason
net	True	True	True	True	NoFailure	NoFailure

- In this example *IPv4Operational State* and *IPv6Operational State* are set to **True** which shows that RSC is enabled for both IPv4 and IPv6

RSC can be disabled and enabled using the GUI or by using PowerShell commands.

Using the GUI

RSC for IPv4 (**Recv Segment Coalescing (IPv4)**) and for IPv6 (**Recv Segment Coalescing (IPv6)**) can be configured using the NIC's advanced properties tab as shown in the example below:



To access the adapter's advanced properties using Windows 2016 (other versions may be slightly different) go to: **Start > Settings > Network and Internet > Change adapter options**, right click the adapter, select **Properties**, click **Configure** and then select the **Advanced** tab.

Using PowerShell

To disable RSC:

```
Disable-NetAdapterRsc -Name "net" -IPv4
```

or for IPv6

```
Disable-NetAdapterRsc -Name "net" -IPv6
```

To enable RSC:

```
Enable-NetAdapterRsc -Name "net" -IPv4
```

or for IPv6

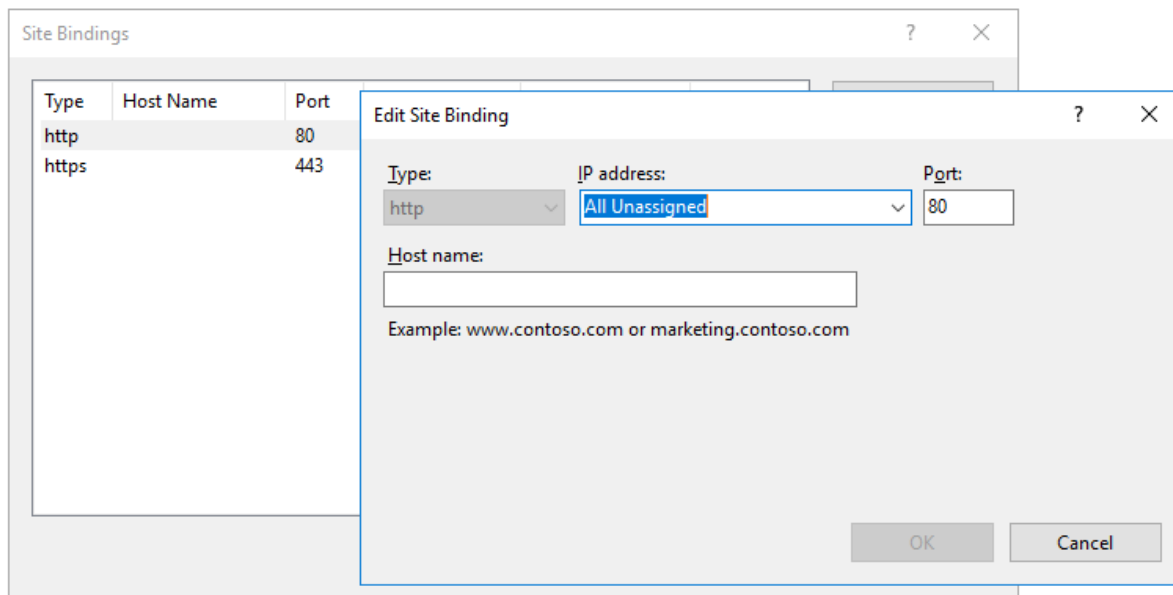
```
Enable-NetAdapterRsc -Name "net" -IPv6
```

- If your network adapter is not named "net" modify the command accordingly

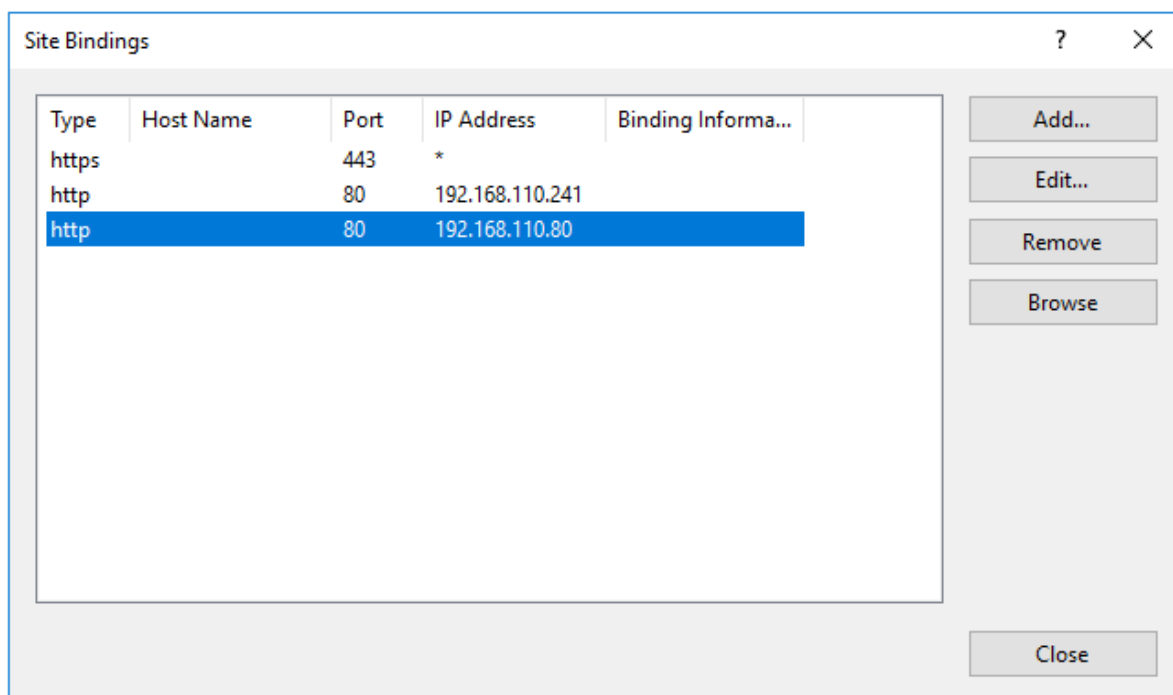
Configuring Your Application to Respond to Both the RIP and VIP

For DR & TUN modes, it's important to make sure that your application (IIS in this example) responds to both the VIP and RIP.

By default, IIS listens on all configured IP addresses as shown in the example below. As can be seen, the IP address field is set to "All Unassigned".



If the default setting is not changed, no further IIS configuration is required. If you do change the IP address in the bindings from "All Unassigned" to a specific IP address, then you need to make sure that you also add a binding for the Virtual Service IP address (VIP) as shown in the example below:



Important This example illustrates how IIS must be configured to ensure that it's listening on both the RIP

and VIP address. Remember that this applies to **ALL** applications when using DR mode.

Windows Firewall Settings

When IIS is installed, default rules are created that allow inbound traffic on port 80 and 443. By default, these rules apply to the Domain, Public and Private profiles and all interface types. If the rules are changed, make sure that inbound traffic can still reach both the LAN interface and the loopback adapter.

NAT Mode Considerations

Layer 4 NAT mode requires Real Server return traffic to pass back via the load balancer. This is achieved by setting each Real Server's default gateway to be the load balancer. For an HA Pair, a floating IP address should be used to allow the gateway to "float" (move) between appliances should a failover occur.

Whilst NAT mode is fairly straight forward, a few points need to be considered.

NAT Mode Potential Issues

1. By default your Real Servers won't be able to access the Internet through the new default gateway (except when replying to requests made through the external VIP).
2. Non-load balanced services on the Real Servers (e.g. RDP for management access to Windows servers) will not be accessible since these have not been exposed via the load balancer.

Enabling Real Server Internet Access Using Auto-NAT

1. Using the WebUI, navigate to: **Cluster Configuration > Layer 4 - Advanced Configuration**.
2. Change **Auto-NAT** from **off** to the **external** interface being used - typically **eth1**.
3. Click **Update**.

This activates the **rc.nat** script that forces external network traffic to be MASQUERADED to and from the external network. The iptables masquerade rule that's used for this is shown below:

```
iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
```

Enabling Access to non Load Balanced Services

If you want non load balanced services on your Real Servers to be accessible from the external subnet, you have two choices:

1. Setup a NAT mode Virtual Service listening on the relevant service ports and add the required Real Server leaving the **Real Server Port** field blank. The Real Server's default gateway must be the load balancer.
2. Setup a floating IP address and add SNAT/DNAT rules for each server as shown in the example below, these lines can be added to the firewall script using the WebUI menu option **Maintenance > Firewall Script**. The Real Server will then be accessible via the floating IP address. Once again, the Real Server's default gateway must be the load balancer.

```
INT_ADDR="192.168.110.240" # The Real Server's IP address
EXT_ADDR="10.200.110.240" # A floating IP on the load balancer in the external subnet
```



```
iptables -t nat -A POSTROUTING -p tcp -s $INT_ADDR -j SNAT --to-source $EXT_ADDR
iptables -t nat -A PREROUTING -p tcp -d $EXT_ADDR -j DNAT --to-destination $INT_ADDR
```

Once configured, firewall entries similar to the following will be listed under *View Configuration > Firewall Rules*:

```
Chain PREROUTING (policy ACCEPT 923 packets, 86608 bytes)
pkts bytes target    prot opt in     out     source           destination
 18   944 DNAT      tcp  --  *      *       0.0.0.0/0        10.200.110.240    to:192.168.110.240

Chain POSTROUTING (policy ACCEPT 23 packets, 1784 bytes)
pkts bytes target    prot opt in     out     source           destination
 0     0 SNAT      tcp  --  *      *       192.168.110.240  0.0.0.0/0        to:10.200.110.240
```

Notes

- INT_ADDR can also be any other server on the internal subnet.
- Separate floating IPs and SNAT/DNAT rules are needed for each internal server.
- The DNAT rule changes the destination IP address of inbound packets from the external floating IP address (EXT_ADDR) to the internal server's address (INT_ADDR).
- The SNAT rule changes the source IP address of outbound packets from the internal server's address (INT_ADDR) to the external floating IP address (EXT_ADDR).
- The default gateway on the internal server must be the load balancer.
- If Auto-NAT is already enabled, only the DNAT rule is required.

One-Arm (Single Subnet) NAT Mode

Normally the VIP is located on a different subnet to the Real Servers. However, it is possible to perform NAT mode load balancing on a single subnet where the VIP is brought up in the same subnet as the Real Servers. For clients located on this subnet, return traffic would normally be sent directly to the client bypassing the load balancer which would break NAT mode. To address this, the routing table on the Real Servers must be modified to force return traffic to pass via the load balancer. The sections below explain how routing must be modified for Windows hosts and Linux hosts.

Route Configuration for Windows Servers

To rectify this issue for Windows servers, a route must be added to each Real Server that takes priority over the default Windows routing rules. This is a simple case of deleting the default On-link route and adding a permanent route via the load balancer using the following commands on each real server:

```
netsh interface ipv4 delete route 192.168.2.0/24 "LAN"
netsh interface ipv4 add route 192.168.2.0/24 "LAN" 192.168.2.21
```

Note

Replace "192.168.2.0/24" with your subnet address, "192.168.2.21" with your load balancer's IP address and "LAN" with the name of the interface on your server.

Note

After running the above commands, reboot the server and check if the updated routing rules have remained. Depending on the specific version of Windows, it may be necessary to add the commands to a startup script. This is because under certain circumstances routing rules for on-link, directly accessible addresses can get reset to defaults after a reboot.

Verify routing rules using the following command:

```
netsh interface ipv4 show route
```

Route Configuration for Linux Servers

To rectify this issue for Linux servers, we need to modify the local network route by changing to a higher metric:

```
route del -net 192.168.2.0 netmask 255.255.255.0 dev eth0
route add -net 192.168.2.0 netmask 255.255.255.0 metric 2000 dev eth0
```

Note

Ensure you specify your local subnet address.

Then we need to make sure that local network access uses the load balancer as its default route:

```
route add -net 192.168.2.0 netmask 255.255.255.0 gateway 192.168.2.21 metric 0 dev eth0
```

Note

Replace 192.168.2.0 & 255.255.255.0 with your local subnet address and replace 192.168.2.21 with the IP address of your load balancer.

Any local traffic (same subnet) is then handled by the manual route and any external traffic is handled by the default route (which also points at the load balancer).

Firewall Marks


Firewall marks are used to group ports and protocols into a single Virtual Service. For example, firewall marks can be used to bundle HTTP connections on port 80 and secure HTTPS connections on port 443 for an e-commerce site. By assigning the same firewall mark to each protocol, state information for the transaction can be preserved because the load balancer forwards all requests from a particular client to the same Real Server.

Firewall Marks - Auto Configuration

When configuring a layer 4 VIP with multiple ports, firewall marks is used automatically in the background to enable this functionality. For example, to configure an HTTP & HTTPS NAT mode Virtual Service, port 80 & 443 must be specified separated by a comma in the Virtual Service's **Ports** field as shown below:

Virtual Service		
Label	<input type="text" value="HTTP-Cluster"/>	?
IP Address	<input type="text" value="192.168.115.100"/>	?
Ports	<input type="text" value="80,443"/>	?
Protocol		
Protocol	<input type="text" value="TCP"/>	?
Forwarding		
Forwarding Method	<input type="text" value="NAT"/>	?


This will automatically configure the required firewall marks.


 **Note** Persistence will be enabled automatically.


When configuring the Real Servers, the **Ports** field must be left blank as shown below:

Label	<input type="text" value="IIS1"/>	?
Real Server IP Address	<input type="text" value="192.168.30.22"/>	?
Real Server Port	<input type="text"/>	?
Weight	<input type="text" value="100"/>	?
Minimum Connections	<input type="text" value="0"/>	?
Maximum Connections	<input type="text" value="0"/>	?

Packets will then be forwarded to the Real Servers on the same port as received by the VIP.

 **Note** For Layer 4 DR mode VIPs, there is no Real Server Port field since port translation is not possible in this mode.

 **Note** To create an auto firewall mark VIP that listens on **all** ports, enter an asterisk (*) in the ports field rather than a specific port number.

 **Note** The health check port is automatically set to the first port in the list. This can be changed if required by modifying the VIP and setting the required port in the **Check Port** field.

Firewall Marks - Manual Configuration






Firewall Marks can also be configured manually. The basic concept is to create a firewall rule that matches incoming packets to a particular IP address/port/protocol and mark them with an arbitrary integer. When the Virtual Service is configured, this firewall mark integer is specified rather than an IP address.

Note

In most cases, it's not necessary to create firewall marks manually. Multiple ports and protocols can easily be dealt with using the standard [Creating Layer 4 Virtual Services](#) feature.

Step 1: Create the New VIP

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 4 - Virtual Services*.
2. Click **Add a new Virtual Service**.

Virtual Service		
Label	<input type="text" value="Cluster-1"/>	
Firewall Mark Identifier	<input type="text" value="1"/>	
Ports	<input type="text" value="80"/>	
Protocol		
Protocol	<input type="text" value="Firewall Marks"/>	
Forwarding		
Forwarding Method	<input type="text" value="Direct Routing"/>	
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

3. Specify the required *Label* (name) for the VIP, e.g. **Cluster-1**.
4. Instead of entering an IP address, enter a numeric value, e.g. **1** - this is the numeric reference for the Firewall Mark, this reference is used in step 5 below when configuring the firewall rules.
5. The *Ports* field does not need to be set as it is not relevant in this case - the actual port(s) used are configured in the firewall script in step 5 below.
6. Set *Protocol* to **Firewall Marks** - at this point the *Ports* field will be grayed out and the *IP Address* field will be renamed as *Firewall Mark Identifier* as shown above.
7. Click **Update**.

Note

Persistence will be enabled automatically.

Step 2: Specify a Health Check Port

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 4 - Virtual Services*.
2. Click **Modify** next to the new Virtual Service.



3. Enter the appropriate value in the *Check Port* field.
4. Click **Update**.

Step 3: Add the Real Servers

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 4 - Real Servers*.
2. Click **Add a new Real Server**.
3. Enter the required details as shown below.

Label	<input type="text" value="Server1"/>	?
Real Server IP Address	<input type="text" value="192.168.111.241"/>	?
Weight	<input type="text" value="100"/>	?
Minimum Connections	<input type="text" value="0"/>	?
Maximum Connections	<input type="text" value="0"/>	?

CancelUpdate

4. Click **Update**.

Step 4: Add the Associated Floating IP Address for the VIP

Note

If you're modifying an existing layer 4 VIP, the floating IP will already exist so this step can be skipped.

1. Using the WebUI, navigate to: *Cluster Configuration > Floating IPs*.
2. Add a floating IP that corresponds to the required VIP, in this example **192.168.111.240**.

New Floating IP

Add Floating IP

3. Click **Add Floating IP**.

Step 5: Modify the Firewall Script

1. Using the WebUI, navigate to: *Maintenance > Firewall Script*.
2. Scroll down to the Manual Firewall Marks section and add the following lines as shown below:

```
VIP1="192.168.111.240"
iptables -t mangle -A PREROUTING -p tcp -d $VIP1 --dport 8025 -j MARK --set-mark 1
```



```
iptables -t mangle -A PREROUTING -p udp -d $VIP1 --dport 8025 -j MARK --set-mark 1
```

Firewall Script

```
32 ##### Manual Firewall Marks #####
33
34 # Example: Associate HTTP and HTTPS with Firewall Mark 1:
35 #VIP1="10.0.0.66"
36 #iptables -t mangle -A PREROUTING -p tcp -d $VIP1 --dport 80 -j MARK --set-mark 1
37 #iptables -t mangle -A PREROUTING -p tcp -d $VIP1 --dport 443 -j MARK --set-mark 1
38
39 # A Virtual Service may then be created in the web interface, using 1 as the
40 # service address.
41
42 #It is also possible to bind TCP and UDP protocols together with a firewall mark.
43 #VIP1="192.168.64.27"
44 #iptables -t mangle -A PREROUTING -p tcp -d $VIP1 --dport 80 -j MARK --set-mark 1
45 #iptables -t mangle -A PREROUTING -p udp -d $VIP1 --dport 300 -j MARK --set-mark 1
46
47 VIP1="192.168.111.240"
48 iptables -t mangle -A PREROUTING -p tcp -d $VIP1 --dport 8025 -j MARK --set-mark 1
49 iptables -t mangle -A PREROUTING -p udp -d $VIP1 --dport 8025 -j MARK --set-mark 1
50
51 ##### Packet Filtering #####
52
53 # You should always use a network perimeter firewall to lock down all
54 # external access to the load balancer except the required Virtual Services
55 # and the required services from your admin machine / network (SSH & HTTPS)
56
57 # Allow unlimited traffic on the loopback interface:
58 #iptables -A INPUT -i lo -j ACCEPT
59 #iptables -A OUTPUT -o lo -j ACCEPT
```

Update

3. Click **Update**.
4. For a clustered pair, make the same changes to the firewall script on the Secondary appliance.

The VIP is now configured and should be accessible on 192.168.111.240, TCP & UDP port 8025.

Firewall Mark Notes

1. When using firewall marks the load balancer forwards traffic to the selected Real Server without changing the destination port. So incoming traffic to port 80 on the Virtual IP will be forwarded to port 80 on one of the Real Servers. Likewise, incoming traffic to port 443 will be forwarded to port 443 on the same Real Server.
2. A single health check port can be configured. For auto-firewall marks this is set as the first port in the list by default. For manual firewall marks this must be manually configured.
3. You can specify a range of ports rather than a single port as shown below:

```
iptables -t mangle -A PREROUTING -p tcp -d 10.141.12.34 --dport 1024:5000 -j MARK --set-mark 1
```

- This specifies destination ports from 1024 to 5000
4. You can leave the upper limit blank to use the default upper limit as shown below:

```
iptables -t mangle -A PREROUTING -p tcp -d 10.141.12.34 --dport 1024: -j MARK --set-mark 1
```



- This specifies destination ports from 1024 to 65535

5. You can specify a range of IP addresses as shown below:

```
iptables -t mangle -A PREROUTING -p tcp -m iprange --dst-range 10.141.12.34-10.141.12.40 --dport 80 -j MARK --set-mark 1
```

- This specifies the destination IP address as a range from 10.141.12.34 to 10.141.12.40

Layer 4 - Advanced Configuration

This section allows you to configure various layer 4 global settings.

Lock Ldirectord Configuration	<input type="checkbox"/>	?
Check Interval	<input type="text" value="5"/>	?
Check Timeout	<input type="text" value="3"/>	?
Negotiate Timeout	<input type="text" value="5"/>	?
TCP FIN Timeout	<input type="text" value="120"/>	?
UDP Timeout	<input type="text" value="300"/>	?
Failure Count	<input type="text" value="2"/>	?
Quiescent	<input type="text" value="no"/>	?
Email Alert Source Address	<input type="text"/>	?
Email Alert Destination Address	<input type="text"/>	?
Auto-NAT	<input type="text" value="off"/>	?
Multi-threaded	<input type="text" value="yes"/>	?

[Update](#)

Lock Ldirectord Configuration - Prevent the WebUI from writing the Ldirectord configuration file, so that manual changes are retained. Manual changes to the Ldirectord configuration file may be overwritten if settings are edited in the WebUI. Locking the configuration file will prevent the WebUI from modifying the file so that custom edits are preserved. A warning message will be displayed on all Layer 4 configuration pages, and changes will be denied.

WARNING: The Layer 4 configuration is set to read-only – changes made on this page will not be saved. Read-only mode may be disabled on the **Advanced Configuration** page.

Check Interval - Layer 4 health check interval in seconds. If this setting is too low, you may experience unexpected Real Server downtime.



Check Timeout - Layer 4 health check timeout in seconds. If this setting is too low, you may induce unexpected Real Server downtime.

TCP FIN Timeout - The time to remember an TCP session for after seeing a FIN packet.

UDP Timeout - The time to remember a session for after seeing a UDP packet. The timeout is reset on every UDP packet received.

Negotiate Timeout - Layer 4 negotiate health check timeout in seconds. The negotiate checks may take longer to process as they involve more server side processing than a simple TCP socket connect check. If this setting is too low, you may induce unexpected Real Server downtime.

Failure Count - Layer 4 number of times a check has to fail before taking server offline. The time to detect a failure and take down a server will be (check interval + check timeout) x failure count.

Quiescent - When a Real Server fails a health check, do we kill all connections?

When Quiescent is set to **yes**, on a health check failure the Real Server is not removed from the load balancing table, but the weight is set to 0. Persistent connections will continue to be routed to the failed server, but no new connections will be accepted. When Quiescent is set to **no**, the server is completely removed from the load balancing table on a health check failure. Persistent connections will be broken and sent to a different Real Server.

Note

Quiescent only applies to health checks - it has no effect on taking Real Servers offline in System Overview. To manually force a Real Server to be removed from the table, set Quiescent to no and arrange for the server to fail its health check. This may be done, for example, by shutting down the daemon or service, changing the negotiate check value, or shutting down the server.

Email Alert Source Address - Specify the global source address of the email alerts. When an email alert is sent, the system will use this address in the "From" field.

Email Alert Destination Address - Specify the global destination email alert address. This address is used to send notifications of Real Server health check failures. This can also be configured on a Virtual Service level.

Auto-NAT - Automatically NAT outbound network connections from internal servers. By default servers behind the load balancer in a NAT configuration will not have access to the outside network. However, clients on the outside will be able to access load balanced services. By enabling Auto-NAT the internal servers will have their requests automatically mapped to the load balancer's external IP address. The default configuration is to map all requests originating from internal network **eth0** to the external IP on **eth1**. If you are using a different interface for external traffic you can select it here. Manual SNAT and DNAT configurations for individual servers can also be configured in the firewall script.

Multi-threaded - Perform health checks with multiple threads. Using multiple-threads for health checks will increase performance when you have a large number of Virtual Services.

Layer 7 Services

The Basics



Layer 7 services are based on HAProxy which is a fast and reliable proxying and load balancing solution for TCP, HTTP/1.1 and HTTP/2 applications.

Since HAProxy is a full proxy, Layer 7 services are not transparent by default, i.e. the client source IP address is lost as requests pass through the load balancer and instead are replaced by an IP address owned by the load balancer. This is the interface IP by default, but can also be set to any other IP address that the load balancer owns, for example the VIP address.

When a VIP is added, the load balancer automatically adds a corresponding floating IP address which is activated instantly. You can use the WebUI option: **View Configuration > Network Configuration** to ensure that the floating IP address is active. It will be listed as a secondary address/alias.

Multiple persistence methods (aka affinity/stickiness) are supported and can be enabled when needed.

Multiple ports can be specified, for example 80 & 443. In this case persistence will probably be required (default setting) to ensure that clients hit the same backend server for both HTTP & HTTPS traffic and also prevent the client having to renegotiate the SSL connection.

With Layer 7, port re-direction is possible, i.e. VIP:80 → RIP:8080 is supported.

Manual configuration of layer 7 services is supported using the WebUI menu option: **Cluster Configuration > Layer 7 - Manual Configuration**. This enables custom layer 7 VIPs to be created that are able to use HAProxy features not directly supported via the WebUI. For more information, please refer to [Layer 7 - Custom Configurations](#).

 **Note**

It's not possible to configure a VIP on the same IP address as any of the network interfaces. This ensures services can "float" (move) between Primary and Secondary appliances when using an HA Pair.

Creating Layer 7 Virtual Services (Using the Wizard)

The wizard can be used to configure Layer 7 Virtual Services and the associated Real Servers.

To run the wizard:

1. Using the WebUI, navigate to: **Cluster Configuration > Setup Wizard** and click **General Layer 7 Virtual Service**.
2. Configure the required Virtual Service settings as shown in the example below:

Setup Wizard - General Layer 7 Virtual Service

Load balancer configuration		
	Primary	Secondary
Hostname	lbmaster	Not configured
Static IP Addresses eth0	192.168.111.200/18	
Floating IP Addresses		

Create a new Layer 7 Virtual Service


Label	<input type="text" value="Web-Cluster"/>		
Virtual Service	IP Address	<input type="text" value="192.168.111.210"/>	
	Ports	<input type="text" value="80"/>	
Layer 7 Protocol	<input type="text" value="HTTP Mode"/>		

[Create Virtual Service](#)

3. Click **Create Virtual Service**.

4. Now continue and add the associated load balanced servers (Real Servers) as shown below:

Attach Real Servers

Label	IP Address	Port	Weight	
<input type="text" value="Web1"/>	<input type="text" value="192.168.111.220"/>	<input type="text" value="80"/>	<input type="text" value="100"/>	
<input type="text" value="Web2"/>	<input type="text" value="192.168.111.221"/>	<input type="text" value="80"/>	<input type="text" value="100"/>	

[Add Real Server](#)

[Attach Real Servers](#)

- Use the **Add Real Server** button to configure additional Real Servers and use the red cross to delete Real Servers.
- Once you're happy, click **Attach Real Servers** to create the Real Servers.
- A confirmation message will be displayed as shown in the example below:

Information: Real Server Web1 added.

Information: Real Server Web2 added.

Information: Virtual Service configured successfully

Continue

- Click **Continue**.
- Finally, reload HAProxy using the **Reload HAProxy** button in the "Commit changes" message box at the top of the screen or by using the WebUI menu option: **Maintenance > Restart Services** and clicking **Reload HAProxy**.



Note

Running the wizard again will permit additional Layer 7 VIPs and associated RIPs to be configured.



Note

By default, Real Server health checks are set to a TCP port connect. If you need to configure a more robust check, please refer to [Chapter 8 - Real Server Health Monitoring & Control](#).

Creating Layer 7 Virtual Services

Virtual services (VIPs) can be configured either by creating a new VIP and configuring all required settings, or by using the duplicate VIP feature to clone an existing VIP and then modifying the settings as needed.

Each Virtual Service can have an unlimited number of Real Servers. Typically you'll need one Virtual Service for each distinct cluster (group of load balanced servers). For example, you'd create a VIP for a web cluster, another for an FTP cluster and a third for a SIP cluster.

Configuring a New Layer 7 VIP

- Using the WebUI, navigate to: **Cluster Configuration > Layer 7 - Virtual Services**.
- Click **Add a new Virtual Service**.

Virtual Service		[Advanced +]
Label	<input type="text" value="VIP Name"/>	?
IP Address	<input type="text" value="10.0.0.20"/>	?
Ports	<input type="text" value="80"/>	?
Protocol		
Layer 7 Protocol	<input type="text" value="HTTP Mode"/>	?
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>



3. Enter an appropriate **Label** (name) for the new Virtual Service.
4. Enter the required IP address in the **IP Address** field.

 **Note**

For the Virtual Service to listen on all IP addresses configured on the appliance (including the management address) specify **0.0.0.0** in the **IP address** field.

5. Enter the required port(s) in the **Ports** field. Individual port numbers should be separated by commas, and ranges may be specified using a dash. For example: 81, 443 - 447, 103, 104, 105. To exclude ports, use a single exclamation mark. Individual ports and ranges can be excluded. For example: 75-90!(80,82) or 300-400!(320-350!330-340) or (40-50,60)!41-44.

 **Note**

The appliance uses a number of ports for various system functions. By default, most are bound to all appliance IPs and therefore cannot be used for Virtual Services unless reconfigured. For details, please refer to [Ports Used by the Appliance](#).

6. Select the required **Layer 7 protocol**:

- **HTTP Mode** - Use the mode if the Virtual Service will handle only HTTP traffic. This allows more flexibility in the processing of connections. When using persistence types such as HTTP Cookie and HTTP Application, HTTP mode must be selected. In addition, the HAProxy log will show more information on the client requests and Real Server responses.
- **TCP Mode** - Use this mode to support all non HTTP traffic such as HTTPS, RPC, RDP, FTP, SIP etc.
- **HTTP/2 Mode** - Use this mode to support HTTP/2.

7. For **HTTP/2 Mode** VIPs:

- Select the required **SSL Certificate**.
- If client validation is required, select the relevant **CA Certificate**.
- Select the required **CA Certificate Verification** level:
 - **Required** - Connections from clients with invalid certificates will be dropped immediately.
 - **Optional** - Connections with an invalid certificate will be allowed to the associated Virtual Service where further actions may be taken using ACL rules. For example, redirecting clients with an invalid certificate to a custom error page.
 - **Never** - No certificate verification will occur. For example, this could be useful if the certificate was signed by a private certificate authority.
- To configure advanced HTTP/2 settings, click **[Advanced]**.
 - Configure the required **Ciphers to use**, the default is:

```
ECDHE-ECDSA-AES256-GCM-SHA384 : ECDHE-ECDSA-AES128-GCM-SHA256 : DHE-RSA-AES256-GCM-SHA384 : DHE-RSA-AES128-GCM-SHA256 : ECDHE-RSA-AES256-GCM-SHA384 : ECDHE-RSA-AES128-GCM-SHA256
```

- Configure the remaining Cipher options.



8. Click **Update**.

9. To configure the Real Servers, please refer to [Creating Layer 7 Real Servers \(RIPs\)](#).

Duplicating an Existing Layer 7 VIP

If you have existing Virtual Services, these can be duplicated using the "Duplicate Service" feature.

Note

This option copies all Virtual Service settings and the associated Real Servers. Once duplicated, you'll need to change either the IP address or port for the new VIP so that it does not clash with the original.


To duplicate an existing layer 7 VIP:

1. Click **Modify** next to the VIP you'd like to duplicate.
2. Click the **Duplicate Service** button.
3. Click **OK** at the prompt to confirm you want to duplicate the VIP.
4. The VIP will be duplicated with a new label, all other settings will be identical.
5. Change the **IP Address**, **Port** and any other setting to suit your requirements.
6. Click **Update**.


Modifying a Layer 7 VIP


When a Virtual Service is first created, only certain settings can be configured. All other settings are set at default values to simplify initial configuration. These settings can be changed after the Virtual Service has been created by clicking **Modify** next to the Virtual Service. the following table details the additional settings that can be changed:

Section	Setting	Description
Virtual Service	[Advanced] > <i>Manual Configuration</i>	Enabling this option will prevent the HAProxy configuration file being written for this Virtual Service, leaving the user to configure it via the Layer 7 - Manual Configuration page. If the Virtual Service label is the same in the manual configuration you will be able to see the status of the Virtual Service as well as control it via the System Overview as you would any other service. <div>Note For more information on creating a manually configured layer 7 VIP, please refer to Layer 7 - Custom Configurations.</div>

Section	Setting	Description
	[Advanced] > Create Backend Only	<p>Enabling this option will prevent the automatic creation of a Frontend in the HAProxy configuration file for this Virtual Service's Backend. The pool of Real Servers will not be directly accessible to clients via the network but can instead be made accessible from another Virtual Service by naming this Virtual Service in a Backend ACL rule.</p> <div>  Note <p>For more information on creating backend VIPs, please refer to HAProxy Backends.</p> </div>
Protocol	[Advanced] > HTTP Pipeline Mode (HTTP mode only)	<p>Select how HAProxy should handle HTTP pipelining to client and server. The options are:</p> <ul style="list-style-type: none"> • Keep-alive Both - Enable pipelining from both the client to HAProxy and from HAProxy to the server. • Close both client and server - Disable pipelining, always closing connections to both client and server after the end of the response, and appending the Connection: close header in both directions. • Keep-alive client, close server - Allow client to negotiate pipelining, whilst closing the server connection using HTTP.
	[Advanced] > Work around broken Connection: close (HTTP mode only)	<p>Work around Real Servers that do not correctly implement the HTTP Connection:close option.</p>
	[Advanced] > Accept Invalid HTTP Requests (HTTP mode only)	<p>This allows invalid characters in header names to be passed through to the backend. If a fix is not immediately available, enable this option. However it can hide further application bugs as well as open security breaches and should only be enabled as a last resort. Ultimately fix your application.</p>
	[Advanced] > HTTP request timeout (DoS Protection) (HTTP mode only)	<p>Enabling this option helps protect against Slowloris type attacks. With this option enabled the client must send the full HTTP header request within 5 Seconds.</p>



Section	Setting	Description
	[Advanced] > <i>Reuse Idle HTTP Connections</i> (HTTP mode only)	It's possible to reuse idle connections to serve requests from the same session which can be beneficial in terms of performance. It's important to note that the first request of a session is always sent over its own connection, and only subsequent requests may be dispatched over other existing connections.
	[Advanced] > <i>TCP Keep-alive</i> (TCP mode only)	Enables the transmission of TCP keep-alive on both the client and the server sides of the connection. It's important to note that this has nothing to do with HTTP keep-alive. This Option is enabled by default when using persistence modes - MS Session Broker and RDP Client Cookie.
	[Advanced] > <i>Redispatch</i>	If a real server becomes un-responsive ignore persistence and send client connection to another available real server. If Unsure leave enabled.
Connection Distribution Method	<i>Balance Mode</i>	<p>The scheduler used to specify server rotation. Specify the scheduler to utilize when deciding the backend server to use for the next new connection. The options are:</p> <ul style="list-style-type: none"> • Weighted Round Robin - Incoming requests are distributed to Real Servers in a sequential manner relative to each Real Server's weight. Servers with a higher weight receive more requests. A server with a weight of 200 will receive 4 times the number of requests than a server with a weight of 50. Weightings are relative, so it makes no difference if Real Server #1 and #2 have weightings of 50 and 100 respectively or 5 and 10. The default weight for new Real Servers is 100. • Weighted Least Connection (default for new VIPs) - Incoming requests are distributed to Real Servers with the fewest connections relative to each Real Server's weight. Servers with a higher weight receive more requests. A server with a weight of 200 will receive 4 times the number of requests than a server with a weight of 50. Again, weightings are relative, so it makes no difference if Real Server #1 and #2 have weightings of 50 and 100 respectively or 5 and 10. The default weight for new Real Servers is 100. <i>This is the default method for new VIPs.</i> • First - The first server with available connection slots receives the connection. The servers are chosen from the lowest numeric identifier to the highest which defaults to the server's position in the farm. Once a server reaches its maxconn value, the next server is used.

Section	Setting	Description
Persistence	Persistence Mode	<p>Select how the load balancer should track clients so as to direct each request to the same server. The options are:</p> <ul style="list-style-type: none"> • HTTP Cookie (HTTP mode only) - The load balancer will set an HTTP Cookie to track each client. • Application Cookie (HTTP mode only) - Where an existing HTTP Cookie is set by the web application on the Real Servers, use this to track each client. • SSL Session ID (TCP mode only) - Read the Session ID from the SSL connection and use this to track each client. • MS Session Broker (TCP mode only) - Use the server-set msts RDP Cookie to track clients connecting to a Microsoft Terminal Server. The Session Broker service must be enabled on the real servers. • RDP Client Cookie (TCP mode only) - Use the client-set mstshash RDP Cookie to track clients connecting to a Microsoft Terminal Server. If the cookie is missing, source IP persistence will be used instead. • Source IP - The same source IP always hits the same Real Server. • HTTP Cookie and Source IP (HTTP mode only) - As HTTP Cookie, falling back to Source IP if the cookie is missing from the HTTP request. • X-Forwarded-For and Source IP (HTTP mode only) - Use X-Forwarded-For, falling back to Source IP if the X-Forwarded-For header is missing from the request. <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p> Note You cannot use the set X-Forwarded-For header option with this method of persistence. It will be disabled.</p> </div> <ul style="list-style-type: none"> • Stick On Fallback – This option disables automatically failing back to the Real Server from the fallback server when the Real Server comes back online. An external fallback server is required. This method is appropriate where you have one Real Server and one fallback server. If the Real Server fails, traffic will be handled by the fallback server. When the Real Server comes back online, the fallback server will continue to handle all traffic and no automatic fallback to the Real Server will occur. In order to force fallback you will need to clear the stick table.


Section	Setting	Description
		<ul style="list-style-type: none"> • Last Successful - Clients will use which ever server last successfully served a client. <div>  Note This includes the fallback server if one is configured. </div> <p>The last successful server is recorded in the stick table and clearing this will cause the server selection for the next connection to be determined by the Balance Mode.</p> <ul style="list-style-type: none"> • None - No persistence. The allocation of clients to Real Servers will be determined solely by the Balance Mode.
Persistence Options	<i>HTTP Cookie Name</i>	Set the name of the HTTP cookie.
	<i>Application Cookie Name</i>	Set the name of the application cookie.
	[Advanced] > <i>HTTP Cookie Max Idle Duration</i>	Set the max idle time of the cookie.
	[Advanced] > <i>HTTP Cookie Max Life Duration</i>	Set the max lifetime of the cookie.
	[Advanced] > <i>HTTP Cookie Attributes</i> > <i>HttpOnly</i>	Appends the HttpOnly attribute to the persistence cookie. Cookies marked this way will be inaccessible to non-HTTP components and, in particular, inaccessible to javascript.
	[Advanced] > <i>HTTP Cookie Attributes</i> > <i>Secure</i>	Appends the Secure attribute to the persistence cookie. Cookies marked this way will only be sent with requests made over secure channels such as HTTPS.
	[Advanced] > <i>HTTP Cookie Domain</i>	Appends one Domain attribute for each hostname in this space-separated list to the persistence cookie. The Domain attribute specifies the domains which this cookie will be sent to.
	[Advanced] > <i>Persistence timeout</i>	The time-out period before an idle connection is removed from the connection table. The source IP address will be removed from memory when it has been idle for longer than the persistence timeout. The default units are minutes.
	[Advanced] > <i>Persistence table size</i>	The size of the table of connections in KB. The size of the table of connections (approximately 50 bytes per entry) where connection information is stored to allow a session to return to the same server within the timeout period. The default units are in KB.

Section	Setting	Description
	[Advanced] > <i>Clear Stick on Drain</i>	Clear the stick table entries for a particular real server when that server is drained using the system overview. Clearing the associated stick table entries when draining a real server is particularly useful and recommended if you have long lived connections with large connection timeouts such as RDP or SSH. When the server is taken out of drain mode and this option is enabled users will be load balanced across all servers in the normal way when they reconnect, when disabled (default) users are reconnected to the same server until their persistence entry in the stick table expires.
	[Advanced] > <i>XFF IP Position</i>	<p>With XFF headers it's possible to have either more than one header or more than one IP in that header. This option gives the user the ability to select a specific IP position inside the header to use for persistence. For example: X-Forwarded-For: 192.168.1.1, 192.168.1.2, 10.10.10.1.</p> <p>In the above example the -1 (default) position is 10.10.10.1 this will always be the last appended value, -2 is 192.168.1.2 and -3 is 192.168.1.1 and so on for as many IPs as you have in your header.</p> <p>It's possible to do the same thing with Multiple XFF headers:</p> <p>X-Forwarded-For: 192.168.1.1</p> <p>X-Forwarded-For: 192.168.1.2</p> <p>X-Forwarded-For: 10.10.10.1</p> <p>It works the same as the previous example -1 is 10.10.10.1 or the most recently added header -2 is 192.168.1.2 and -3 is 192.168.1.1 and so on. The IP address at the position you select will be stored in the stick table and used for persistence on the next request from the user.</p>

Section	Setting	Description
Health Checks	Check Type	<p>Specify the type of health check to be performed on the Real Servers. The options are:</p> <ul style="list-style-type: none"> • Negotiate HTTP/HTTPS (GET) - Scan the page specified in Request to Send, and check the returned data for the Response Expected string • Negotiate HTTP/HTTPS (HEAD) - Request the page headers of the page specified in Request to Send • Negotiate HTTP/HTTPS (OPTIONS) - Request the options of the page specified in Request to Send • Connect to port - Attempt to make a connection to the specified port. • External script - Use a custom file for the health check. For more information, please refer to External Health Check Scripts. • MySQL - The check consists of sending two MySQL packets, one Client Authentication packet, and one QUIT packet, to correctly close the MySQL session. It then parses the MySQL Handshake Initialization packet and/or Error packet. It's a basic but useful test and does not produce error nor aborted connect on the server. However, it requires adding an authorization in the MySQL table, like this: <div data-bbox="608 1064 1414 1126" data-label="Text"> <pre>USE mysql; INSERT INTO user (Host,User) values ('',''); FLUSH PRIVILEGES;</pre> </div> • No checks, Always on - No health checks, all real servers are marked online. <div data-bbox="584 1341 616 1379" data-label="Image"> </div> <div data-bbox="584 1341 699 1379" data-label="Section-Header"> <h4>Note</h4> </div> <div data-bbox="801 1326 1409 1395" data-label="Text"> <p>For full details of all layer 7 health check options, please refer to Health Checks for Layer 7 Services.</p> </div>
ACL Rules		Enables ACLs to be configured. For more information on configuring ACLs, please refer to ACLs (Access Control Lists) .
Header Rules (HTTP mode only)		Enables HTTP headers to be added, set or deleted. For more information on configuring HTTP headers, please refer to Modifying HTTP Header Fields .

Section	Setting	Description
Feedback Method	<i>Feedback Method</i>	<p>Select whether HAProxy should query each Real Server for its load level. The options are:</p> <ul style="list-style-type: none"> • None - HAProxy will not modify the Real Server's weight. • Agent - The Real Server is queried every health check interval for the real server's percent CPU idle. This is used to set each Real Server's weight to a value proportional to its initial weight. For example, if the initial weight is 100 and the percentage CPU idle is 34, the weight will be set to 34. Remember lower numbers mean lower priority for traffic, when compared with other real servers in the pool.
Fallback Server	<i>Disable Fallback</i>	<p>Disable fallback server functionality. In the case that all real servers are failing health checks connections will not be sent to a fallback server, instead TCP connections will be closed and HTTP requests will be answered with a 503.</p>
	<i>IP Address</i>	<p>The server to route to if all of the Real Servers fail their health check. By default the fallback server is set to be the local NGINX instance.</p> <div>  Note For more information on configuring the fallback server, please refer to Fallback Server. </div> <p>You can also configure the fallback server to be a "hot spare" if required. To do this, configure the primary server as a Real Server and set the secondary server as the fallback server.</p>
	<i>Port</i>	<p>The port of the fallback server. By default this is set to 9081.</p>
	[Advanced] > <i>Fallback Persistence</i>	<p>Configure the Fallback server to be persistent. During a health check failure users can be forwarded to a fallback server. Setting this to on will make this server persistent so that when the Real Servers are put back in the pool, they will remain on the fallback server until their persistence times out.</p> <p>When disabled (default), users will be moved to a Real Server when one becomes available.</p> <div>  Note There is a delay equal to the current setting of the "Slow Start Delay" (<i>Cluster Configuration > Layer 7 - Advanced</i>) before users are moved to a Real Server. </div>
	[Advanced] > <i>Encrypt Connection</i>	<p>Enable SSL encryption to the fallback server.</p>
SSL	<i>Enable Backend Encryption</i>	<p>Enabling this option will enable by default the use of HTTPS for all new Backend Servers. This options can then be disabled per backend server under the Real Server settings.</p>

Section	Setting	Description
Other	[Advanced] > <i>Maximum Connections</i>	Specifies the maximal number of concurrent connections that will be sent to this server. If the number of incoming concurrent requests goes higher than this value, they will be queued, waiting for a connection to be released.
	[Advanced] > <i>Timeout</i>	<p>Use this option to override the default client & server timeouts in the Layer 7 advanced section.</p> <p>Client Timeout - The inactivity timeout applies when the client is expected to acknowledge or send data.</p> <p>Real Server Timeout - The inactivity timeout applies when the server is expected to acknowledge or send data.</p>
	[Advanced] > <i>Set X-Forwarded-For Header</i>	Instruct HAProxy to add an X-Forwarded-For (XFF) header to all requests, showing the client's IP Address. If HTTP is selected under Layer 7 Protocol, HAProxy is able to process the header of incoming requests. With this option enabled, it will append a new X-Forwarded-For header containing the client's IP Address. This information may be extracted by the Real Server for use in web applications or logging.
	[Advanced] > <i>Force to HTTPS</i>	<p>If set to "Yes" any HTTP connections that are made on this VIP will be forced to reconnect using HTTPS. This will keep any entered URL. If you are terminating the SSL on the Load balancer you should use the same VIP address for both the SSL Termination and Layer 7 configurations.</p> <p>HTTPS Redirect Code - this indicates which type of HTTP redirection is desired. Codes 301, 302, 303, 307 and 308 are supported, with 302 used by default if no code is specified. The options are:</p> <ul style="list-style-type: none"> • 301 means "Moved permanently", and a browser may cache the Location. • 302 means "Moved permanently" and means that the browser should not cache the redirection. • 303 is equivalent to 302 except that the browser will fetch the location with a GET method. • 307 is just like 302 but makes it clear that the same method must be reused. • 308 replaces 301 if the same method must be used. <p>HTTPS Redirect Port - Setting this option to a port other than 443 will cause a port to be specified in the Force-to-HTTPS redirection emitted when clients connect via HTTP.</p>
	[Advanced] > <i>Use RIP name as Host Header</i>	When set to "Yes" the Host Header is set to the name allocated to the RIP.

Section	Setting	Description
	[Advanced] > <i>Force SSL Content Inspection</i>	Insert the rules to inspect the ClientHello packet for SSL/TLS connections even if the rules are not required for the configuration.
	[Advanced] > <i>Accept Proxy Protocol</i>	<p>If you wish to use this VIP with STunnel for SSL off-load or another supported proxy such as Amazons ELB whilst passing the client's IP address to the real servers this option needs to be enabled (checked). If using with STunnel please ensure that Enable Proxy Protocol is enabled in your STunnel VIP.</p> <div>  Note <p>When used with STunnel, the preferred method is to use the Enable Proxy Protocol option in the STunnel VIP's configuration in conjunction with the Bind Proxy Protocol to L7 VIP option. This will configure both the STunnel VIP and the HAProxy VIP in a single step and allows a single HAProxy VIP to support both HTTP and HTTPS. For more information, please refer to Transparency at Layer 7.</p> </div>
	[Advanced] > <i>Send Proxy Protocol</i>	<p>Enable Proxy Protocol to the backend servers. This option allows the back end servers to see the client's IP address. It should only be enabled if your server supports Proxy Protocol and is configured to use it. The options are:</p> <ul style="list-style-type: none"> • Send V1 - This uses the first version of the Proxy Protocol and send the headers in a human readable format. • Send V2 - This is the newer version of the Protocol and sends the headers in binary. • Send V2 SSL - This is used to show the client was connected over SSL/TLS. • Send V2 SSL CN - This is the same as V2 SSL but also provides the Common Name from the client certificate if set.
	[Advanced] > <i>Enable Compression</i>	Enable gzip HTTP compression. The following MIME types will be compressed when this is enabled: text/html , text/plain , text/css , text/xml , text/javascript , application/javascript , application/xml
	[Advanced] > <i>Set Source Address</i>	Allows the setting of the source IP address that your backend server will see the traffic coming from. This is useful when you wish to only allow a known IP Address to access your Real Servers or need to allow access through a public gateway.
	[Advanced] > <i>Enable HSTS</i>	HSTS specifies a period of time during which the users browser (agent) should only access the server in a secure fashion. The recommended duration should be three months or more.
	[Advanced] > <i>Tunnel Timeout</i>	Timeout for the websocket protocol tunnel when no data is passed between client and server. Can be specified as s/m/h for seconds/minutes/hours.

Section	Setting	Description
	[Advanced] > <i>Transparent Proxy</i>	This will set the source address of the data stream leaving the load balancer destined for the real server to be that of the client. As a result you will need to set the default gateway for the real servers to be that of the load balancer. Other wise, depending on the source address the return traffic will route round the load balancer and no responses will be received. Note: Enabling this feature will enable the TProxy system. Once enabled if no longer required it will need to be disabled from: <i>Layer7 - Advanced Configuration > Transparent Proxy</i> .



Tip

If you require a custom gateway for a particular VIP, this can be achieved using Policy Based Routing. For more information, please refer to [Policy Based Routing \(PBR\)](#).

ACLs (Access Control Lists)

The WebUI supports the ability to create ACLs which can be used to control and direct traffic based on a set of defined rules. This functionality is also known as URL rewriting, URL redirection, URL forwarding or content switching. This option can be accessed under the **ACL Rules** section by clicking the **Add Rule** button when modifying a VIP:

ACL Rules

Type	Bool	URL/Text	Action	Redirect
<div>Add Rule ?</div>				

This brings up the ACLs pop-up menu:

HAProxy

ACL Rule:

Cancel
Ok

Type
Select

Select the desired ACL type from the **Type** dropdown list, fill in the details as appropriate, and create the ACL by clicking the **Ok** button.

HAProxy

ACL Rule:

Cancel
Ok

Type
path_beg

Bool
Equals

URL/Text
/example

Action
Drop

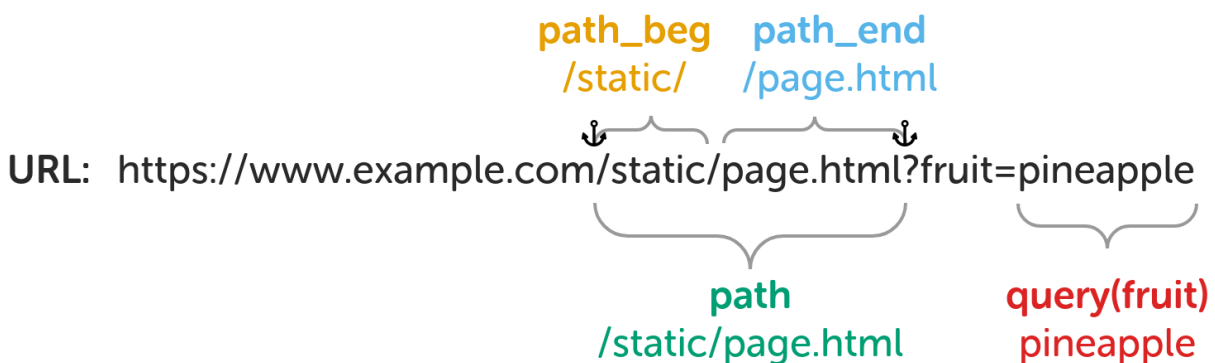
An ACL can be modified by clicking on it in the ACL list. ACLs can be reordered by clicking and dragging them in the ACL list:

ACL Rules				
Type	Bool	URL/Text	Action	Redirect
header(Content-Type)	Equals	application/json	Drop	Remove
hdr_host	Equals	example.com	Drop	Remove
hdr_beg(Host)	Equals	www	Drop	Remove

Different types of ACLs can be created. Some ACLs are dependent on information from the application layer and so are only available for *HTTP mode* virtual services. The ACL types and their supported modes are listed below.

ACL Type	TCP Mode Support	HTTP Mode Support
path	✗	✓
path_beg	✗	✓
path_end	✗	✓
hdr	✗	✓
hdr_host	✗	✓
hdr_beg(Host)	✗	✓
Query	✗	✓
SNI	✓	✗
Flags	✓	✓
IP Address	✓	✓
Port	✓	✓
Free Type	✓	✓

URL-Based ACLs



path:

- Match against the request's full URL path, which starts at the first slash and ends before the (optional) question mark (signifying the start of the query string).
- **Bool:**
 - **Equals:** Perform action if *URL/Text* value matches.
 - **Not equal:** Perform action if *URL/Text* value does not match.
- **URL/Text:** String to compare the full URL path to, e.g. `static/page.html`.
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

path_reg:

- Match against a regular expression (regex).
- **Bool:**
 - **Equals:** Perform action if *URL/Text* value matches.
 - **Not equal:** Perform action if *URL/Text* value does not match.
- **URL/Text:** Regex to match, e.g. `^\/blog[^\/*]*$` - match if the trailing slash after "blog" is missing.
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

path_beg:

- Match against the *beginning* of the request's URL path.
- **Bool:**
 - **Equals:** Perform action if *URL/Text* value matches.
 - **Not equal:** Perform action if *URL/Text* value does not match.
- **URL/Text:** String to compare the beginning of the URL path to, e.g. `/static/` or `/level_1/level_2/`.
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

path_end:

- Match against the *end* of the request's URL path.
- **Bool:**
 - **Equals:** Perform action if *URL/Text* value matches.
 - **Not equal:** Perform action if *URL/Text* value does not match.
- **URL/Text:** String to compare the end of the URL path to, e.g. `/page.html` or `.png`.
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

Query (aka url_param):



- Match against a specified query string parameter.
- **Header/Param:** Name of the query string parameter to match against, e.g. `fruit`.
- **Bool:**
 - **Equals:** Perform action if *URL/Text* value matches.
 - **Not equal:** Perform action if *URL/Text* value does not match.
- **URL/Text:** String to compare the specified query string parameter to, e.g. `pineapple`.
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

Header-Based ACLs

hdr_beg(host)
www.

URL: `https://www.example.com/static/page.html?fruit=pineapple`

hdr(host)
www.example.com

Header: `Content-Type: application/json`

hdr(Content-Type)
application/json

hdr_host (aka `hdr(host)`):

- Match against the request's full Host header.
- **Bool:**
 - **Equals:** Perform action if *URL/Text* value matches.
 - **Not equal:** Perform action if *URL/Text* value does not match.
- **URL/Text:** String to compare the full Host header to, e.g. `www.example.com` or `en.wikipedia.org`.
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

hdr_beg(host):

- Match against the *beginning* of the request's Host header.
- **Bool:**
 - **Equals:** Perform action if *URL/Text* value matches.
 - **Not equal:** Perform action if *URL/Text* value does not match.



- **URL/Text:** String to compare the beginning of the Host header to, e.g. `www.` or `en.`
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

hdr:

- Match against a specified request header.
- **Header/Param:** Name of the request header to match against, e.g. `Content-Type`.
- **Bool:**
 - **Equals:** Perform action if *URL/Text* value matches.
 - **Not equal:** Perform action if *URL/Text* value does not match.
- **URL/Text:** String to compare the specified request header to, e.g. `application/json`.
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

Network/Transport Layer-Based ACLs

IP Address (aka src):

- Match against the source IP address of the request.
- **Bool:**
 - **Equals:** Perform action if *URL/Text* value matches.
 - **Not equal:** Perform action if *URL/Text* value does not match.
- **URL/Text:** IP address to compare the request's source IP address to. CIDR notation can be used and multiple comma-separated values can be given, e.g. `10.0.0.0/8` `123.45.67.8`.
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

Port (aka dst_port):

- Match against the destination TCP port of the request.
- **Bool:**
 - **Equals:** Perform action if *URL/Text* value matches.
 - **Not equal:** Perform action if *URL/Text* value does not match.
- **URL/Text:** Integer to compare the request's destination TCP port to.
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

Miscellaneous ACLs

SNI:

- Match against the Server Name Indication (SNI) TLS extension field of the request.
- **Bool:**



- **Equals:** Perform action if *URL/Text* value matches.
- **Not equal:** Perform action if *URL/Text* value does not match.
- **URL/Text:** String to compare the SNI field to, e.g. `www.example.com`.
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

Flags:

- Match based on the status of flags set by other ACL rules.
- **Bool:**
 - **Equals:** Perform action if condition described by *URL/Text* field evaluates to *true*.
 - **Not equal:** Perform action if condition described by *URL/Text* field does not evaluate to *true*.
- **URL/Text:** Condition to test, e.g. `flag_a || flag_b` or `protected_path internal_src_addr`.
- **Action:** Action to perform if condition is met. See [ACL Actions](#).

Note

See [ACL Examples](#) for an example of how to use the flags feature.

Free Type:

- Free-form custom ACL rule(s) to write into the HAProxy configuration verbatim.
- **Freetype:** ACL configuration line(s) to write into the HAProxy configuration file.

Note

Multiple ACLs can be defined in a single block. Enter one ACL per line.

Tip

Full and detailed documentation on how to write ACLs can be found in the HAProxy Configuration Manual, [here](#).

ACL Actions

When an ACL rule matches an action is taken (or, alternatively, an action is taken when the rule *doesn't* match, depending on the "bool" setting of the rule). Some actions are dependent on information from the application layer and so are only available for *HTTP mode* virtual services. The different actions and their supported modes are listed below.

ACL Type	TCP Mode Support	HTTP Mode Support
Drop	✓	✓
Deny	✗	✓
Set Flag	✓	✓
URL Location	✗	✓
URL Prefix	✗	✓

ACL Type	TCP Mode Support	HTTP Mode Support
Use Backend	✓	✓
Use Server	✓	✓

- **Drop** (aka reject): Stop and immediately close the connection without sending a response.
- **Deny**: Stop and immediately deny the request, emitting the chosen HTTP status code as a response.
 - **Status code**: Status code to use as a response.
 - 200 OK
 - 400 Bad Request
 - 403 Forbidden
 - 405 Method Not Allowed
 - 408 Request Timeout
 - 425 Too Early
 - 429 Too Many Requests
 - 500 Internal Server Error
 - 502 Bad Gateway
 - 503 Service Unavailable
 - 504 Gateway Timeout
- **Set Flag**: Set a flag for use in subsequent ACL rules of type "Flag".
 - **Location/Value**: Name of the flag to set, e.g. `flag_a` or `protected_path`.
- **URL Location** (aka redirect location): Redirect the request to the exact location specified, using a 301 Moved Permanently status code.
 - **Location/Value**: Exact location to redirect to, e.g.
https://en.wikipedia.org/wiki/User_Datagram_Protocol.
- **URL Prefix** (aka redirect prefix): Redirect the request to the *URL path* originally requested prefixed with a specified string, using a 301 Moved Permanently status code.
 - **Location/Value**: String to prefix to the requested URL path to create the redirect location, e.g.
<https://www.example.com>.
- **Use Backend**: Use the specified backend, or another virtual service, to handle the request.
 - **Location/Value**: Name of a valid backend, or another virtual service, to use, e.g.
[apache_srv_cluster_b](#).
- **Use Server**: Use the specified server to handle the request.
 - **Location/Value**: Name of a valid real server, in the same virtual service or backend, to use, e.g.
[apache_srv_7](#).

ACL Examples

Example 1

A virtual service occasionally sees requests for a retired domain, `www.foo.com`. These requests need to be redirected to the new domain: `www.bar.com`. For example, a request for `https://www.foo.com/static/diagram.svg` must be redirected to `https://www.bar.com/static/diagram.svg`.

ACL type to use: *hdr_host*, matching against `www.foo.com`.

ACL action to use: *URL Prefix*, with the prefix `https://www.bar.com`.

Example 2

A web service has been moved: previously, all of its resources were located under `/web-service/`, but now everything is located under `/legacy/web-service/`. Any requests for old locations, whose paths start with just `web-service`, must be redirected to the correct new locations.

ACL type to use: *path_beg*, matching against `/web-service/`.

ACL action to use: *URL Prefix*, with the prefix `/legacy`.

Example 3

A web service hosts an administration panel which is located under `/admin/`. The only legitimate use of this panel should be from users on the local network, which is 10.0.0.0/8. Any non-local users attempting to access the administration panel should be redirected to a branded page, located at `https://example.com/restricted.html`, which explains that they have attempted to access a restricted part of the service.

First ACL type to use: *path_beg*, matching against `/admin/`.

First ACL action to use: *Set Flag*, setting the flag `is_admin_panel`.

Second ACL type to use: *IP Address*, matching against the network 10.0.0.0/8.

Second ACL action to use: *Set Flag*, setting the flag `is_local_user`.

Third ACL type to use: *Flags*, matching against `is_admin_panel !is_local_user`.

Third ACL action to use: *URL Location*, with the location `https://example.com/restricted.html`.

The two flag names together, `is_admin_panel !is_local_user`, create a logical AND (a logical OR could be achieved, instead, by explicitly placing `||` between the flag names). The exclamation mark negates the match on `is_local_user`. The resulting expression of the third ACL will match, and cause the redirection action to be



carried out, when `is_admin_panel` is **true** and `is_local_user` is **false**.

(!) Important The ACL that evaluates the two flags **must** be placed **after** the two ACLs that set the flags.

Example 4

It's necessary to force the use of a particular real server in some scenarios. This must be achieved by setting a particular query string parameter to the name of the server. For example, `http://192.168.0.10/?server_override=apache_srv_dev` should trigger the override ACL condition and send the request to the special server.

ACL type to use: *Query*, looking for the `server_override` parameter and matching against `apache_srv_dev`.

ACL action to use: *Use Server*, with the server name `apache_srv_dev`.

Example 5

You have the following VIPs, both with source IP persistence enabled:

- VIP1 that listens on 192.168.10.10:80 and forwards traffic to the Real Servers on port 8080
- VIP2 that listens on 192.168.10.10:443 and forwards traffic to the same Real Servers on port 4443

You want all requests from a particular client to be handled by the same Real Server irrespective of which VIP receives the request. You can't group port 80 and 443 into a single VIP because port translation is used. To achieve the same objective, both VIPs are configured to share the same persistence table.

Configuration steps:

1. Set VIP2's *Persistence Mode* to **None**.
2. Create an ACL for VIP2 with the following settings:
 - type to use: *Free Type*
 - value: **stick on src table VIP1**

HAProxy Backends

When an ACL is created and the *Action* is set to **Use Backend** it's possible to set the *Location/Value* field to either a backend only VIP, a standard VIP or a manually defined VIP.

1 - Backend only VIP (Recommended)

A Backend VIP does not have a frontend in the HAProxy configuration, only a backend. As a result, there is no floating IP address associated with the VIP. This kind of VIP is used exclusively for ACLs where access is only needed from another VIP and not directly from clients over the network.

To create a backend only VIP:



1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Virtual Services*.
2. Click **Add a New Virtual Service**.
3. Click **[Advanced]** and enable (check) the *Create Backend Only* checkbox.
4. Enter a suitable *Label* (name), e.g. **backend-1**.
5. Select the required *Layer 7 Protocol*.
6. Click **Update**.
7. Now navigate to: *Cluster Configuration > Layer 7 - Real Servers*.
8. Click the **Add a New Real Server** button next to the newly created VIP.
9. Define all the Real Servers that make up the backend.

This new backend VIP can now be referenced as **backend-1** in ACLs.

Note

You can modify the Backend Virtual Service in the normal way if additional settings must be configured.

2 - As a Manually Defined Backend

Manually defined backends allow additional custom settings that are not directly supported via the WebUI to be configured.

To create a manually defined backend:

Using the WebUI menu option: *Cluster Configuration > Layer 7 - Manual Configuration*, the backend "Blog" can be defined as shown below:

```
backend Blog
mode http
balance roundrobin
option forwardfor
server rip3 192.168.110.242:80 weight 1 check
server rip4 192.168.110.243:80 weight 1 check
```

3 - As a Standard VIP with the Required Real Servers

Standard VIPs can also be used. Here, "Blog" has been defined as an additional standard VIP with two Real Servers:

	Blog	192.168.112.116	80	0	HTTP	Layer 7	Proxy	
	REAL SERVER	IP	PORTS	WEIGHT	CONNS			
↑	BlogRIP1	192.168.110.240	80	100	0	Drain	Halt	
↑	BlogRIP2	192.168.110.243	80	100	0	Drain	Halt	

 **Note**

When defining ACLs that have their **Action** set to **Use Backend** or **Use Server**, the relevant Backend VIP or Real Server must exist before HAProxy can be successfully restarted. Note also that names used are case sensitive.

 **Note**

For more details on configuring ACLs please also refer to the HAProxy online documentation available [here](#).

Using Regular Expressions to Rewrite Requests

Regular expressions can be used to rewrite HTTP requests. This is often used to maintain compatibility between old and new URLs or to turn user-friendly URLs into CMS-friendly URLs, etc. This is achieved using the "reqrep" and "reqrep" keywords within a manual layer 7 VIP. The following examples illustrate how these commands can be used:

Example 1

Replace "/static/" with "/" at the beginning of any request path.

```
reqrep ^([^\ :]*)\ /static/(.*) \1\ /\2
```

Example 2

Replace any host name in the HTTP header with "www.mywebsite.com".

```
reqrep ^Host:\ Host:\ www.mysite.com
```

Example 3

Replace /jpg/ with /images/ while maintaining the components before and after the folder.

```
reqrep ^([^\ ]*)\ /jpg/(.*) \1\ /images/\2
```

 **Note**

HAProxy uses PCRE compatible regular expressions. For more information about PCRE syntax, see [Regex Quick Start](#) and [Regex Cheat Sheet](#).

 **Note**

The "reqrep" keyword is strictly case-sensitive, while "repirep" is case insensitive. For more details on configuring manually defined layer 7 VIPs, please refer to [Configuring Manual Virtual Services](#).

Modifying HTTP Header Fields

For **HTTP Mode** Virtual Services, the WebUI supports the ability to add, set, delete, and replace HTTP header fields. This option can be accessed under the **Header Rules** section by clicking the **Add Rule** button when modifying a VIP:

Header Rules				
Phase	Action	Header	Value	Flags
<div> Add Rule ? </div>				

This brings up the headers pop-up menu:

HAProxy

Header Rule:

Cancel

Ok

Type

Request

▼

Option

Select

▼

Select the desired header type from the *Type* dropdown list, fill in the details as appropriate, and create the header rule by clicking the **Ok** button.

HAProxy

Header Rule:

Cancel

Ok

Type

Request

▼

Option

Set

▼

Header

Via-Loadbalancer

Value

Yes

Flags

A header rule can be modified by clicking on it in the header rules list. Header rules can be reordered by clicking and dragging them in the header rules list:

Header Rules				
Phase	Action	Header	Value	Flags
Request	Set	Via-Loadbalancer	Yes	Remove
Request	Delete	Forwarded		Remove
Request	Delete	TE		Remove
<div> Add Rule ? </div>				

Two different types of header rules can be created:



- **Request:** Affect specified HTTP header fields in HTTP *request* messages.
- **Response:** Affect specified HTTP header fields in HTTP *response* messages.

Four different header field manipulation options are supported:

Option	Description
Add	Append an HTTP header field. If a header field of the same name already exists then an additional header field will still be appended.
Set	Append an HTTP header field. If a header field of the same name already exists then it is first removed before a new one is appended. This is useful for handling security information which external users must not be able to set themselves.
Delete	Remove all HTTP header fields of a specified name.
Replace	Perform a regular expression powered "find and replace" operation on all HTTP header field values of a specified name.

The specifics of the header field manipulation options are as follows:

Add:

- **Header:** Field name of the HTTP header to append.
- **Value:** Field value of the HTTP header to append.
- **Flags** (optional): Conditionally execute the header manipulation rule based on the status of flags set by ACL rules, e.g. `is_external_request`.

Set:

- **Header:** Field name of the HTTP header to append.
- **Value:** Field value of the HTTP header to append.
- **Flags** (optional): Conditionally execute the header manipulation rule based on the status of flags set by ACL rules, e.g. `is_external_request`.

Delete:

- **Header:** Field name of the HTTP header to delete.
- **Flags** (optional): Conditionally execute the header manipulation rule based on the status of flags set by ACL rules, e.g. `is_external_request`.

Replace:

- **Header:** Field name of the HTTP header to replace.
- **Value:** "Find and replace" operation to execute, of the form `<matching-regular-expression><replacement>`.



- **Flags** (optional): Conditionally execute the header manipulation rule based on the status of flags set by ACL rules, e.g. `is_external_request`.

Note

See [HTTP Header Field Modification Examples](#) for an example of how to use the replace option.

Tip

It's possible to include dynamic information in header fields, for example the IP address of a request. This is achieved by using log format variables, e.g. `%ci` for the client IP address, and sample expressions, e.g. `[%fe_name]` for the name of the virtual service handling the request. This functionality is explained in detail in the HAProxy online documentation: the relevant section can be found [here](#).

HTTP Header Field Modification Examples

Example 1

A secure deployment requires that the X-Forwarded-For header field in client requests never be trusted. The header is under the control of the client and could potentially contain misinformation set by a malicious client. All header fields of this name should be deleted from incoming requests.

Header rule type to use: *Request*.

Header rule option to use: *Delete*.

Header rule "Header" value to use: *X-Forwarded-For*.

Example 2

A poorly designed web service behind the load balancer leaks sensitive information through an HTTP response header field. The vulnerable response field looks like so: `Database-Engine: MongoDB_3.4.14`. All header fields of this name should be deleted from outgoing responses.

Header rule type to use: *Response*.

Header rule option to use: *Delete*.

Header rule "Header" value to use: *Database-Engine*.

Example 3

A web service behind the load balancer expects to receive information about client requests via HTTP header fields. The service expects to receive the source IP address, destination IP address, and destination port of the client's initial connection to the load balancer, which it expects to find in header fields named X-Client-Source, X-Client-Dest, and X-Client-Dest-Port, respectively. The load balancer should add these header fields to incoming requests and populate their values appropriately. The load balancer should also delete any pre-existing header fields that use the field names that the web service is logging, to prevent clients from tampering with or injecting arbitrary data into the web service's logs.

First header rule type to use: *Request*.



First header rule option to use: **Set**.

First header rule "Header" value to use: **X-Client-Source**.

First header rule "Value" value to use: **%ci**.

Second header rule type to use: **Request**.

Second header rule option to use: **Set**.

Second header rule "Header" value to use: **X-Client-Dest**.

Second header rule "Value" value to use: **%fi**.

Third header rule type to use: **Request**.

Third header rule option to use: **Set**.

Third header rule "Header" value to use: **X-Client-Dest-Port**.

Third header rule "Value" value to use: **%fp**.

Example 4

Any requests containing the HTTP header field "Fruit" with a corresponding value of "pineapple" should have this value changed to "kiwi".

Header rule type to use: **Request**.

Header rule option to use: **Replace**.

Header rule "Header" value to use: **Fruit**.

Header rule "Value" value to use: **pineapple kiwi**.

Creating Layer 7 Real Servers (RIPs)

You can add an unlimited number of Real Servers to each Virtual Service. Real Servers in a Layer 7 configuration can be on any subnet/network provided they are accessible from the load balancer.

To add a new layer 7 RIP:

1. Using the WebUI, navigate to: **Cluster Configuration > Layer 7 - Real Servers**.
2. Click **Add a new Real Server** next to the relevant Virtual Service.



Label	<input type="text" value="RIP Name"/>	?
Real Server IP Address	<input type="text"/>	?
Real Server Port	<input type="text"/>	?
Re-Encrypt to Backend	<input type="checkbox"/>	?
Enable Redirect	<input type="checkbox"/>	?
Weight	<input type="text" value="100"/>	?

- Enter an appropriate **Label** (name) for the new Real Server.
- Enter the required IP address in the **Real Server IP Address** field.
- Enter the required port in the **Real Server Port** field.

Note For a multiport VIP, this field should be left blank.

Note A valid FQDN can be used rather than an IP address if required. The name is resolved using the DNS server(s) specified in **Local Configuration > Hostname & DNS**.

- Enable the **Re-Encrypt to Backend** option if required.

Note For more details about this option, please refer to [SSL Termination on the Load Balancer with Re-encryption \(SSL Bridging\)](#) and [Mutual Transport Layer Security \(mTLS\)](#).

- If required, enable the **Enable Redirect** option. When enabled, this option allows a particular Real Server to respond to GET or HEAD requests with a redirect to a specified location. Requests will be redirected to a URL made up of the prefix specified and the path of their GET or HEAD request. Other request types will continue to be handled by the Real Server.

Note This option is only available when the associated VIP's **Layer 7 Protocol** is set to **HTTP Mode**.

- Specify the required **Weight**, this is an integer specifying the capacity of a server relative to the others in the pool, valid values are 0 to 256, the default is 100. The higher the value, the more connections the server will receive. If the weight is set to 0, the server will effectively be placed in drain mode.

Note To configure **Minimum Connections** and **Maximum Connections**, modify the Real Server after it has been created, these options will then be available.

- Click **Update**.

Layer 7 - Custom Configurations

Custom, manually configured Layer 7 services are useful when your configuration requires advanced HAProxy settings that are not directly supported by the WebUI.

Configuring Manually Defined Virtual Services

Often, the best approach is to first create a VIP and its associated RIPs in the normal way, configuring as many settings as you can using the available WebUI configuration options, then convert this to a manual VIP and add the custom HAProxy configuration settings that you require. The steps are as follows:

- **Step 1** - Define the VIP in the normal way using the WebUI menu option: *Cluster Configuration > Layer 7 - Virtual Services*.
- **Step 2** - Define the required Real Servers in the normal way using the WebUI menu option: *Cluster Configuration > Layer 7 - Real Servers*.
- **Step 3** - Using the WebUI menu option: *View Configuration > Layer 7*, scroll down to the newly created VIP and copy the whole configuration section for the VIP - from the initial **Listen <VIP name>** line, right to the end of the Real Server definitions.
- **Step 4** - Click **Modify** next to the newly created VIP, then click **[Advanced]** in the *Virtual Service* section. Enable (check) the *Manual Configuration* checkbox and click **Update**.
- **Step 5** - Navigate to: *Cluster Configuration > Layer 7 - Manual Configuration* and paste the configuration copied in Step 3 into the editor window, then add the additional custom configuration settings for the VIP and click **Update** when complete.



Note

When you click update in step 5, a syntax check will be done to ensure that the lines you have added are valid.

Manual Config Example 1 - Simple HTTP Redirect

This example illustrates how a VIP can be created, converted to a manual VIP and then modified to include a custom configuration that forces requests that start with **/staff/** or **/staff** to be redirected to **https://login.domain.com**.



Note

This example is for demonstration purposes only. Complex ACLs can be configured in the WebUI without the need for a manual configuration. For more information on configuring ACLs, please refer to [ACLs \(Access Control Lists\)](#).

The lines that need to be manually inserted to achieve this are:

```
acl ACL-1 path_beg /staff/ (see note 1)
acl ACL-2 path_beg /staff (see note 1)
redirect location https://login.domain.com if ACL-1 or ACL-2 (see note 2)
```

Notes

1. These lines configure two ACLs named **ACL-1** & **ACL-2** where the criteria for a match is that the URL starts with either **/staff/** or **/staff**.



2. This line causes a redirect to **https://login.domain.com** to occur when either ACL is matched.

Configuration Steps:

1. Using the WebUI menu option: **Cluster Configuration > Layer 7 - Virtual Services** create a Layer 7 VIP with the required **Label** (name), **IP Address** and **Port**. At this point leave the **Manual Configuration** checkbox unchecked, e.g.

Virtual Service		[Advanced +]
Label	<input type="text" value="VIP1"/>	?
IP Address	<input type="text" value="192.168.2.110"/>	?
Ports	<input type="text" value="80"/>	?
Protocol		
Layer 7 Protocol	<input type="text" value="HTTP Mode"/>	?
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

2. Using the WebUI menu option: **Cluster Configuration > Layer 7 - Real Servers** configure the associated RIPs in the normal way, e.g.

Label	<input type="text" value="RIP1"/>	?
Real Server IP Address	<input type="text" value="192.168.2.111"/>	?
Real Server Port	<input type="text" value="80"/>	?
Re-Encrypt to Backend	<input type="checkbox"/>	?
Enable Redirect	<input type="checkbox"/>	?
Weight	<input type="text" value="100"/>	?
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

3. Using the WebUI menu option: **View Configuration > Layer 7** scroll down to the newly created VIP. Now copy the entire configuration for the VIP.

```
listen VIP1
bind 192.168.2.110:80 transparent
mode http
balance leastconn
cookie SERVERID maxidle 30m maxlife 12h insert nocache indirect
server backup 127.0.0.1:9081 backup non-stick
option http-keep-alive
timeout http-request 5s
option forwardfor
timeout tunnel 1h
option redispatch
```

```
option abortonclose
maxconn 40000
option httplog
server RIP1 192.168.2.111:80 weight 100 cookie Rip1 check inter 4000 rise 2 fall 2 slowstart
8000 minconn 0 maxconn 0 on-marked-down shutdown-sessions
server RIP2 192.168.2.112:80 weight 100 cookie Rip1 check inter 4000 rise 2 fall 2 slowstart
8000 minconn 0 maxconn 0 on-marked-down shutdown-sessions
```

- Using the WebUI menu option: **Cluster Configuration > Layer 7 - Virtual Services**, modify the VIP and check the **Manual Configuration** checkbox and click **Update**.
- Select the WebUI menu option: **Cluster Configuration > Layer 7 - Manual Configuration** and paste the VIP's configuration into the editor window, then add the extra manual config lines:

```
listen VIP1
bind 192.168.2.110:80 transparent
mode http
balance leastconn
acl ACL-1 path_beg /staff/
acl ACL-2 path_beg /staff
redirect location https://login.domain.com if ACL-1 or ACL-2
cookie SERVERID maxidle 30m maxlife 12h insert nocache indirect
server backup 127.0.0.1:9081 backup non-stick
option http-keep-alive
timeout http-request 5s
option forwardfor
timeout tunnel 1h
option redispatch
option abortonclose
maxconn 40000
option httplog
server RIP1 192.168.2.111:80 weight 100 cookie Rip1 check inter 4000 rise 2 fall 2 slowstart
8000 minconn 0 maxconn 0 on-marked-down shutdown-sessions
server RIP2 192.168.2.112:80 weight 100 cookie Rip1 check inter 4000 rise 2 fall 2 slowstart
8000 minconn 0 maxconn 0 on-marked-down shutdown-sessions
```

- Click **Update**.
- Now reload HAProxy using the **Reload HAProxy** button in the "Commit changes" message box at the top of the screen.

Transparency at Layer 7

HAProxy, Pound and STunnel are all proxies which means that a new connection is established from the proxy out to the backend server in response to an inbound client connection to the proxy. This means that the source IP address of the packet reaching the Real Servers will not be the client's IP address, but an IP address owned by the load balancer. The source IP address applied depends on which proxy is in operation:

HAProxy - By default the IP address of the network interface is used, but this can also be configured to be any IP address that the load balancer owns using the **Set Source Address** field of the Layer 7 VIP.

STunnel - By default the IP address of the STunnel Virtual Service is used, but this can also be configured to be any IP address that the load balancer owns using the **Set Source Address** field of the STunnel VIP.



Pound - The IP address of the network interface is used.

Enabling Transparency

The load balancer can provide the actual client IP address to the Real Servers in two ways:

1. By inserting a header that contains the client IP source address. For HTTP traffic the **X-Forwarded-For (XFF)** header is used, for TCP traffic the **Proxy Protocol Header** is used.

Note

For more information on XFF headers, please refer to [Mozilla - X-Forwarded-For](#), for more information on Proxy Protocol headers, please refer to [HAProxy Technologies - The PROXY Protocol](#).

2. By modifying the Source Address field of the IP packets and replacing the IP address of the load balancer with the IP address of the client. The load balancer uses **TProxy** for this purpose.

Note

In many cases, option 1 (using headers) can be used to achieve your objectives. Option 1 is easier to implement because there are no network topology requirements.

These methods can be used independently or in combination to achieve a range of objectives as illustrated in the [Configuration Examples](#) section.

Inserting Headers

X-Forwarded-For (XFF) Headers

X-Forward-For headers are inserted by HAProxy when the layer 7 VIP option **Set X-Forwarded-For header** is enabled (the default for new layer 7 VIPs). A new X-Forwarded-For header is appended by the load balancer containing the client's IP address. This information can then be extracted by the Real Servers for use in web applications or logging.

Proxy Protocol Headers

STunnel & HAProxy can be configured for Proxy Protocol Headers as described below:

STunnel - To configure STunnel to **send** Proxy Protocol Headers, the STunnel Virtual Service option **Enable Proxy Protocol** must be enabled.

Note

If **Associated Virtual Service** is set when configuring the STunnel Virtual Service **Enable Proxy Protocol** will be enabled by default and cannot be disabled.

HAProxy - To configure HAProxy to **send** Proxy Protocol Headers, the layer 7 Virtual Service dropdown **Send Proxy Protocol** must be set to the required header version/type.

To configure HAProxy to **receive** Proxy Protocol Headers, two methods can be used:

1. By specifying the Layer 7 Virtual Service where the STunnel VIP will forward its connections when creating / modifying the STunnel Virtual Service. This will also automatically configure the layer 7 VIP to expect Proxy Protocol Headers **only** for connections from the STunnel VIP where the option was enabled. In this way, the layer 7 VIP will accept traffic with Proxy Protocol Headers from the STunnel VIP as well as standard traffic



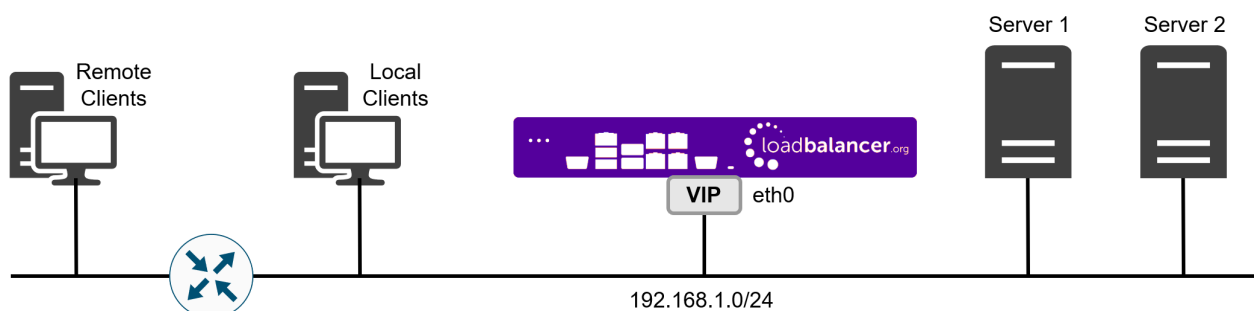
from other sources that do not present Proxy Protocol Headers. To configure this method:

- Click **Modify** next to the STunnel VIP.
 - Set the *Associated Virtual Service* dropdown to the relevant Layer 7 VIP.
 - Click **Update**.
2. By enabling the layer 7 Virtual Service option *Accept Proxy Protocol* - this will configure the layer 7 VIP to expect Proxy Protocol Headers for **all** connections. With this method, the layer 7 VIP will **only** accept connections from sources that present Proxy Protocol Headers. To configure this method:
- Click **Modify** next to the HAProxy VIP.
 - Scroll down to the **Other** section and click **[Advanced]**.
 - Enable (check) *Accept Proxy Protocol*.
 - Click **Update**.

Using TProxy to modify the Source IP Address

Loadbalancer.org appliances utilize TProxy to modify the source IP address of each packet. TProxy can be used in conjunction with HAProxy and Pound. When TProxy is enabled, it's important to be aware of the topology requirements for TProxy to operate correctly. Both one-arm and two-arm topologies are supported:

TProxy Topology Requirements - One-arm Deployments



- Here, the VIP is brought up in the same subnet as the Real Servers.
- To support remote clients, the default gateway on the Real Servers must be an IP address on the load balancer and routing on the load balancer must be configured so that return traffic is routed back via the router.

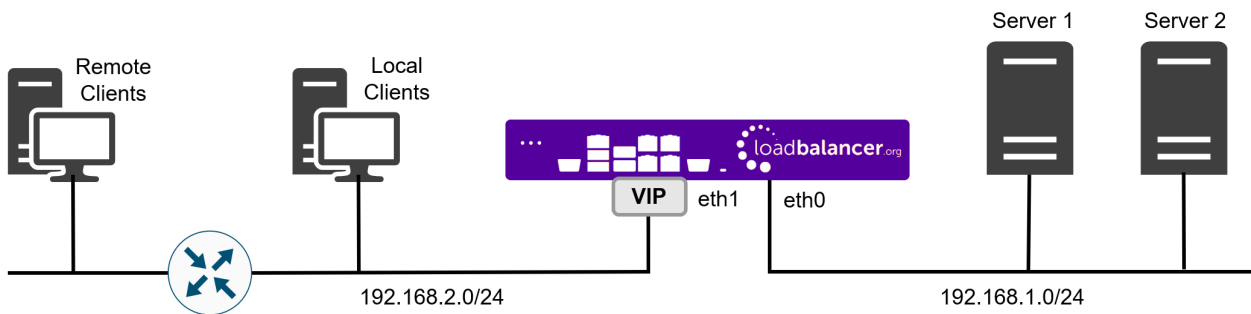
Note

For an HA clustered pair, a floating IP should be added to the load balancer and used as the Real Server's default gateway. This ensures that the IP address can "float" (move) between Primary and Secondary appliances.

- To support local clients, return traffic would normally be sent directly to the client bypassing the load balancer which would break TProxy. To address this, the routing table on the Real Servers must be modified to force return traffic to go via the load balancer in the same way as one-arm NAT mode. For more information, please refer to [One-Arm \(Single Subnet\) NAT Mode](#).

TProxy Topology Requirements - Two-arm Deployments





- Here, two subnets are used. The VIP is located in one subnet and the load balanced Real Servers are located in the other. The load balancer requires two interfaces, one in each subnet.

Note

This can be achieved by using two network adapters, or by creating VLANs on a single adapter.

- The default gateway on the Real Servers must be an IP address on the load balancer.

Note

For an HA clustered pair, a floating IP should be added to the load balancer and used as the Real Server's default gateway. This ensures that the IP address can "float" (move) between Primary and Secondary appliances.

- Clients can be located in the same subnet as the VIP or any remote subnet provided they can route to the VIP.

To enable TProxy for a particular layer 7 VIP:

- Click **Modify** next to the HAProxy VIP.
- Scroll down to the **Other** section and click **[Advanced]**.
- Enable (check) *Transparent Proxy*.
- Click **Update**.

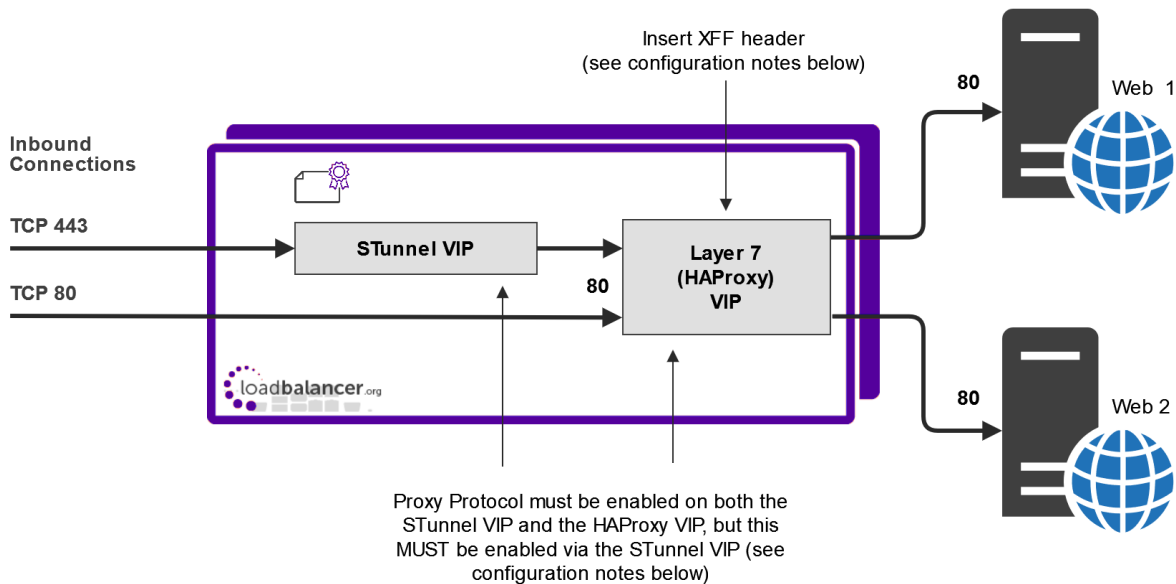
Configuration Examples

Example 1 - Using Proxy Protocol & X-Forwarded-For Headers

In this example, Proxy Protocol Headers are used with STunnel and HAProxy to present the original client source IP address to the load balanced servers in an XFF header inserted by HAProxy.

Note

HAProxy can also be used to perform SSL termination. In this case, STunnel and Proxy Protocol Headers are not needed. For more information on using HAProxy for SSL termination, please refer to [SSL Termination on the Load Balancer \(SSL Offloading\)](#).



Configuration Notes

1. Configure the STunnel VIP and the HAProxy VIP on the same IP address. Clients then connect to a single IP address for HTTP and HTTPS.
2. Proxy Protocol must be enabled via the STunnel VIP, **not** via the Layer 7 (HAProxy) VIP. In this way, the HAProxy VIP where STunnel forwards its traffic is automatically configured to accept traffic **with** Proxy Protocol Headers from the STunnel VIP, and also standard traffic **without** Proxy Protocol Headers from other sources, i.e. the direct HTTP connections.

Configuring the STunnel VIP:

Label	SSL-VIP1	?
Associated Virtual Service	VIP1	?
Virtual Service Port	443	?
SSL Operation Mode	High Security	
SSL Certificate	Default Self Signed Certificate	?
Source IP Address		?
Enable Proxy Protocol	<input checked="" type="checkbox"/>	?
Bind Proxy Protocol to L7 VIP	VIP1	?

Cancel
Update

The STunnel VIP option **Associated Virtual Service** must be set to the backend HAProxy VIP where STunnel will forward its traffic. Then, both the STunnel VIP and the associated HAProxy VIP will be configured automatically.

These STunnel settings will:

1. Configure the STunnel VIP to send Proxy Protocol Headers.
2. Configure the HAProxy VIP to expect Proxy Protocol Headers only from traffic that comes from the STunnel VIP.

Configuring the Layer 7 (HAProxy) VIP:

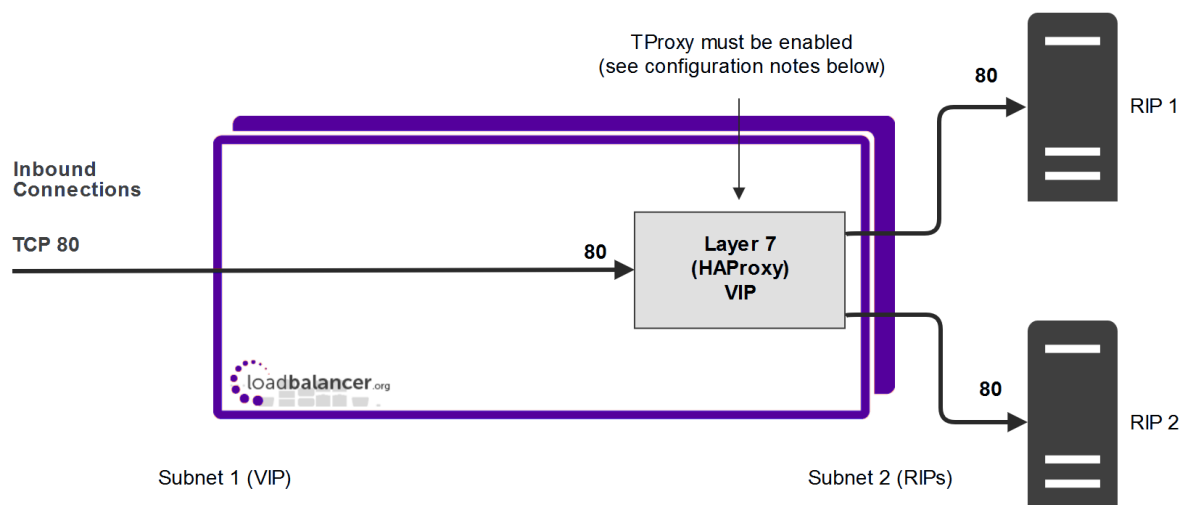
Other		[Advanced -]
Maximum Connections	<input type="text" value="40000"/>	?
Timeout	<input type="checkbox"/>	?
Set X-Forward-For header	<input checked="" type="checkbox"/>	?

X-Forwarded-For Headers must be enabled for HAProxy (this is the default setting).

Once all settings are configured, the **X-Forwarded-For** header received by the load balanced servers Web 1 & Web 2 will contain the source IP address of the client.

Example 2 - Using HAProxy & TProxy

In this example, TProxy is enabled for HAProxy so that the source IP address in IP packets is modified by the load balancer to be the client's IP address.



Configuration Notes & Topology Requirements

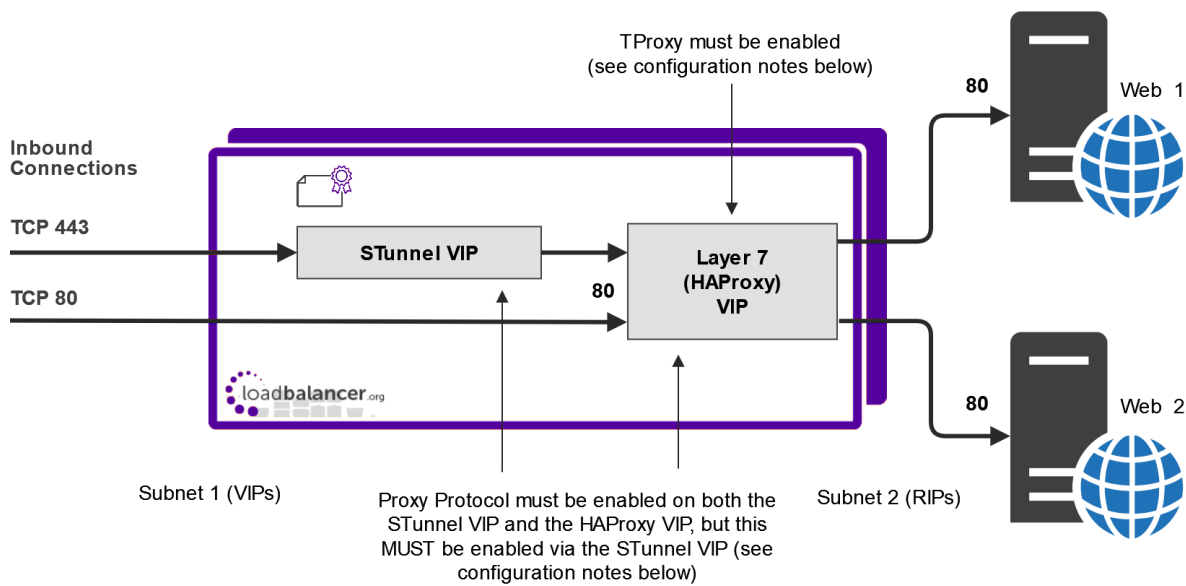
1. Certain topology requirements must be met when using TProxy. For details, please refer to [Using TProxy to modify the Source IP Address](#).
2. TProxy for HAProxy must be enabled. This is done at the VIP level rather than globally as in previous versions. To enable TProxy at the VIP level, click **Modify** next to the VIP in question, scroll down to the **Other** section and click **[Advanced]**, then enable (check) **Transparent Proxy**.
3. On the Real Servers, the default gateway must be configured to be an IP address on the load balancer. When using a clustered pair, this should be a floating IP to allow failover to the Secondary.

Example 3 - Using STunnel, HAProxy & TProxy

In this example, Proxy Protocol Headers are used to pass the client IP address from STunnel to the Layer 7



HAProxy VIP. TProxy is enabled for HAProxy so that the source IP address in packets sent to the Real Servers is modified by the load balancer to be the client's IP address.

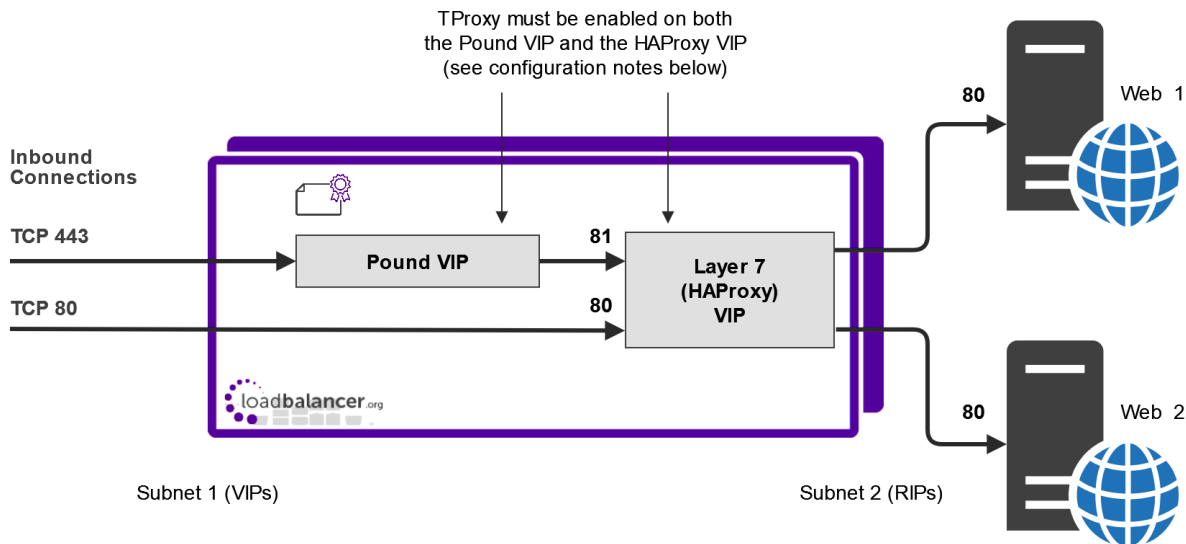


Configuration Notes & Topology Requirements

1. Certain topology requirements must be met when using TProxy. For details, please refer to [Using TProxy to modify the Source IP Address](#).
2. Configure the STunnel VIP and the HAProxy VIP on the same IP address. Clients then connect to a single IP address for HTTP and HTTPS.
3. TProxy for HAProxy must be enabled. This is done at the VIP level rather than globally as in previous versions. To enable TProxy at the VIP level, click **Modify** next to the VIP in question, scroll down to the **Other** section and click **[Advanced]**, then enable (check) **Transparent Proxy**.
4. On the Real Servers, the default gateway must be configured to be an IP address on the load balancer. When using a clustered pair, this should be a floating IP to allow failover to the Secondary.
5. Proxy Protocol **must** be enabled via the STunnel VIP, not via the Layer 7 (HAProxy) VIP. This is done by checking the **Enable Proxy Protocol** option when either creating or modifying the STunnel VIP (please refer to configuration example 1). This automatically configures the HAProxy VIP to accept traffic with Proxy Protocol Headers from the STunnel VIP and also standard traffic from other sources (i.e. the direct HTTP connections) that do not present Proxy Protocol Headers.
6. If you want to enable HTTP to HTTPS redirection, enable **Force to HTTPS** on VIP1.

Example 4 - Using Pound, HAProxy & TProxy

In this example, TProxy is enabled for HAProxy and Pound so that the source IP address is modified by the load balancer to be the client's IP address.



Configuration Notes & Topology Requirements

1. Certain topology requirements must be met when using TProxy. For details, please refer to [Using TProxy to modify the Source IP Address](#).
2. Configure the Pound VIP and the HAProxy VIP on the same IP address. Clients then connect to a single IP address for HTTP and HTTPS.
3. Configure the Layer 7 HAProxy VIP to listen on two ports - e.g. 80 & 81, then use port 80 for client connections on HTTP and port 81 for the Pound backend.
4. When configuring Real Servers for HAProxy VIP, ensure that the **Real Server Port** field is set and not left blank.
5. TProxy for HAProxy must be enabled. This is done at the VIP level rather than globally as in previous versions. To enable TProxy at the VIP level, click **Modify** next to the VIP in question, scroll down to the **Other** section and click **[Advanced]**, then enable (check) **Transparent Proxy**.
6. TProxy for Pound must be enabled using the WebUI menu option: **Cluster Configuration > SSL - Advanced Configuration** and **Transparent Proxy** to On.
7. On the load balanced backend Servers, the default gateway must be configured to be an IP address on the load balancer. When using a clustered pair, this should be a floating IP to allow failover to the Secondary appliance.
8. If you want to enable HTTP to HTTPS redirection, you'll need to split the Layer 7 HAProxy VIP into two separate VIPs, one on port 80 with **Force to HTTPS** enabled and the other configured to accept traffic from Pound.

Layer 7 - Advanced Configuration

This section allows you to configure the various layer 7 global settings.

Lock HAProxy Configuration (Deprecated) - Prevent the WebUI writing to the HAProxy configuration file. Manual changes to the HAProxy configuration file may be overwritten if settings are edited in the WebUI. Locking the configuration file will prevent the WebUI from modifying the file, so that custom edits are preserved. A warning message will be displayed on all Layer 7 configuration pages, and changes will be denied.



Note

Layer 7 manual configurations should be used instead. For more information, please refer to [Layer 7 - Custom Configurations](#).

Logging - Set the required logging level for layer 7 services. Logs are written to /var/log/haproxy.log.

Redispatch - Allows HAProxy to break persistence and redistribute to working servers should failure occur. Normally this setting should not require changing.

Connection Timeout - HAProxy connection timeout in milliseconds. This setting should normally not require changing.

Client Timeout - HAProxy client timeout in milliseconds. This setting should normally not require changing.

Real Server Timeout - HAProxy Real Server timeout in milliseconds. This setting should not require changing.

Maximum Connections - HAProxy maximum concurrent connections. This setting should not require changing, unless you are running a high volume site. See also Maximum Connections for a Virtual Service (HAProxy).

Abort on Close - Abort connections when users close their connection. Recommended as the probability for a closed input channel to represent a user hitting the browser's "stop" button is close to 100%.

Transparent Proxy - Enable TProxy support for Layer 7 HAProxy. TProxy support is required in order for the Real Servers behind a layer 7 HAProxy configuration to see the client source IP address. The load balancer must be in a NAT configuration (internal and external subnets) with the Real Servers using an IP address on the load balancer (preferably a floating IP) as their default gateway. Can be used on its own or in combination with Pound and TProxy.



Note

TProxy must be enabled at the VIP level. This is a change from previous versions where enabling it here would enable it for ALL layer 7 VIPs. To enable TProxy at the VIP level, click Modify next to the VIP in question, scroll down to the Other section and click **[Advanced]**, then enable (check) Transparent Proxy. Setting this at the VIP level will also automatically set the Transparent Proxy option here. For more information on using TProxy, please refer to [Transparency at Layer 7](#).



Note

Since the load balancer must be in a NAT configuration (i.e. VIPs & RIPs in different subnets and default gateway on the Real Servers set as an IP on the load balancer) to utilize TProxy, it's not always an appropriate solution. In situations such as this, it's also possible to use the X-forwarded-for header with layer 7 Virtual Services. Most web servers can then be configured to record the X-Forwarded-For IP address in the log files.



Note

For details on how to enable X-Forwarded-For headers, please click [here](#). For details on how to enable X-Forwarded-For headers in Apache, please refer to our [Apache blog](#), For details on how to enable X-Forwarded-For headers in IIS, please refer to our [IIS blog](#).

Disable On Start - HAProxy brings up all real servers in the UP state after the restart. Enabling this option will bring the real servers up in MAINT mode stopping any connections to them. The init script will then return the real servers back to their previous state pre reload/restart. The init script can do this without this option enabled but while waiting for the init script to get to each service to set the state the real server will be accepting traffic. So it's



recommended that you use this with large deployments, or if you just want to stop connections before the the previous state has been returned.

Interval - Interval between health checks. This is the time interval between Real Server health checks in milliseconds.

Rise - Number of health checks to Rise. The number of positive health checks required before re-activating a Real Server.

Fall - Number of health checks to Fall. The number of negative health checks required before deactivating a Real Server.

Slow Start Time - To minimize the thundering heard effect of a real server recovering from a health check failure getting overwhelmed with all its old users attempting to reconnect at once. This timer will gradually increase the connections for a period set by this value until the end of the timer is reached at which point the server will be running at normal capacity.

 **Note**

If the feed back agent is enabled, the slowstart time MUST be greater than the Interval value.

Feedback Agent Interval - The time in milliseconds between each feedback agent check from HAProxy to the feedback agent.

Advanced Stats - Enable/disable additional actions available on the HAProxy stats page.

Auto-refresh Stats Page - Enable to allow the layer 7 stats page to refresh periodically. Automatic refreshing can be paused and resumed using **Disable refresh** and **Enable refresh** in the **Display options** on the stats page.

Auto-refresh Interval - Enter the time to wait before refreshing after the statis page is loaded. A suffix of us, ms, s, m, h or day can be added to specify microseconds, milliseconds, seconds, minutes, hours or days respectively. The default unit is seconds.

Request Buffer Length - Set the health check buffer length in bytes.

 **Note**

Changing this value will effect the performance of HAProxy. Do not make changes unless you know exactly what you are doing.

Lower values allow more sessions to coexist in the same amount of RAM, and higher values allow some applications with very large cookies to work. The default value is 16384 bytes. It is strongly recommended not to change this from the default value, as very low values will break some services such as statistics, and values larger than the default size will increase memory usage, possibly causing the system to run out of memory. Administrators should consider reducing the Maximum Connections parameter if the request buffer is increased.

Header Buffer Length - Set the header buffer length, in bytes The header buffer is a section of the request buffer, reserved for the addition and rewriting of request headers. The default value is 1024 bytes. Most applications will only require a small header buffer, as few headers are added or rewritten.

Persistence Table Replication - When enabled, HAProxy's persistence tables are replicated to the Secondary

device.

Replication Port - Set the TCP port to use for persistence table replication. The default port is TCP 7778.

eMail Alert From - Set the "from address" for email alerts.

eMail Alert To - Set the "to address" for email alerts.

eMail Server Address - Set the email server address as either an IP address or FQDN.

eMail Server Port - Set the email server TCP port.

Note

For more information on configuring email alerts, please refer to [Configuring Email Alerts for Virtual Services](#).

Enable Multi-threading - This can improve performance if limits are being reached.

Default Number of Threads - Let the appliance choose a sensible number of worker threads. By default this will be the same as the number of cores available when HAProxy starts or reloads.

Number of Threads - Be aware that starting too many threads will have a detrimental affect on performance. Leaving this field blank will have the same effect as selecting default number of threads.

Note

Multi-threading is enabled by default and the number of threads is auto set based on the number of detected CPUs / vCPUs.

Enable Prometheus Exporter - Enable the Prometheus exporter for HAProxy. The end point will be mapped to /ladmin/stats/l7prometheus/ in the WebUI.

HAProxy Status Codes

For reference, HAProxy's may emit the following status codes:

Code	When/Reason
200	access to stats page, and when replying to monitoring requests
301	when performing a redirection, depending on the configured code
302	when performing a redirection, depending on the configured code
303	when performing a redirection, depending on the configured code
307	when performing a redirection, depending on the configured code
308	when performing a redirection, depending on the configured code
400	for an invalid or too large request
401	when an authentication is required to perform the action (when accessing the stats page)
403	when a request is forbidden by a "block" ACL or "reqdeny" filter



Code	When/Reason
404	when the requested resource could not be found
408	when the request timeout strikes before the request is complete
410	when the requested resource is no longer available and will not be available again
500	when haproxy encounters an unrecoverable internal error, such as a memory allocation failure, which should never happen
502	when the server returns an empty, invalid or incomplete response, or when an "rspdeny" filter blocks the response.
503	when no server was available to handle the request, or in response to monitoring requests which match the "monitor fail" condition
504	when the response timeout strikes before the server responds

For a complete HAProxy reference please refer to the [HAProxy Configuration Manual](#).

Floating IPs

In order for the load balancer to function, the appliance must physically own the Virtual IP address that the clients are accessing before they get re-directed to a Real Server in the cluster. When new layer 4 or layer 7 Virtual Services (VIPs) are created, corresponding Floating IPs are added automatically and can be viewed using the WebUI menu option: **Cluster Configuration > Floating IPs**.

It's also possible to manually configure floating IPs if required, this is normally only required when manually configuring firewall marks or when using layer 4 NAT mode or TProxy where in both cases the load balancer must be the default gateway for the Real Servers. A floating IP is required for an HA pair to allow the gateway address to be brought up on the secondary appliance should a failover occur.

Floating IPs are controlled by heartbeat to ensure that only the active appliance (normally the Primary) owns the Floating IP(s) at any time.

To manually add a floating IP:

1. Using the WebUI, navigate to: **Cluster Configuration > Floating IPs**.

Floating IPs

192.168.111.42	Disable	Delete
192.168.111.40	Disable	Delete

New Floating IP

Add Floating IP

- Specify the new floating IP.
- Click **Add Floating IP**.

Note

When using a clustered pair, ensure that the Secondary also has a static IP address assigned that's in the same subnet as the floating IP being added. Failure to do so will result in heartbeat issues during a failover.

Note

To disable a floating IP address and bring the IP down, use the relevant **Disable** button. The **Disable** button will be replaced with an **Enable** button to bring it back up when required.

Note

Floating IPs are not deleted automatically when Virtual Services are removed or the IP address is changed, this must be done manually.

SSL Termination

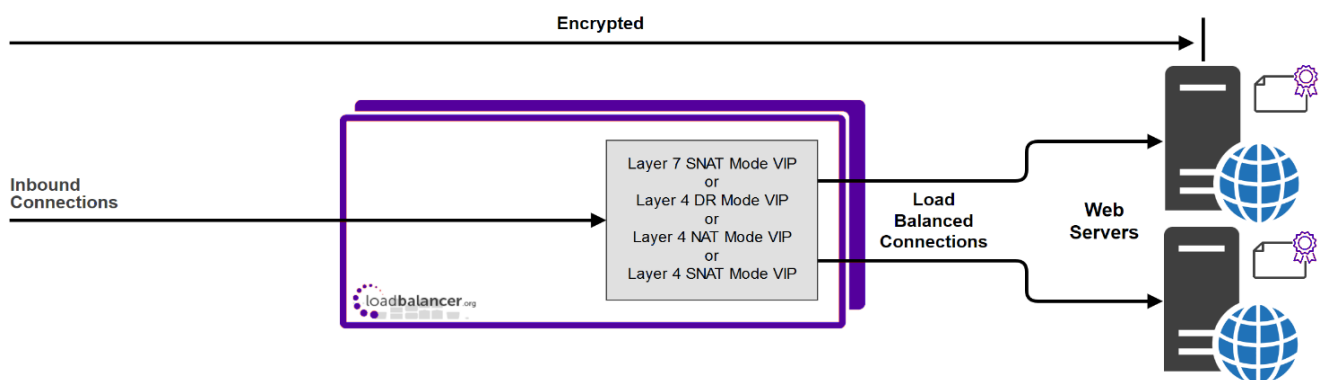
Concepts

SSL termination can be handled in the following ways:

- On the Real Servers (recommended) - aka **SSL Passthrough**.
- On the load balancer - aka **SSL Offloading**.
- On the load balancer with re-encryption to the backend servers - aka **SSL Bridging**.

The following sections describe each method.

SSL Termination on the Real Servers (SSL Passthrough)



In this case SSL certificates are installed on each Real Server in the normal way. Data is encrypted from client to server. This provides full end-to-end data encryption as shown in the diagram above.

Notes

- This is our recommended solution. SSL termination on the load balancer (SSL Offload) can be very CPU intensive and in most cases, for a scalable solution, terminating SSL on the Real Servers is the best option.



- It's not possible to use HTTP cookie persistence as well as other layer 7 techniques that control how traffic is sent to the Real Servers because all data is encrypted as it passes through the load balancer.

The load balancer is configured with a VIP that listens on HTTPS port 443 and distributes inbound requests to the Real Servers on port 443 as shown below:

SSL	192.168.110.50	Port 443/tcp	Direct Routing	Add a new Real Server	
SSL1	192.168.110.51	443	Weight 100	Modify	Delete
SSL2	192.168.110.52	443	Weight 100	Modify	Delete

A fairly common configuration is to include port 80 in the VIP's definition and also enable persistence. This ensures that both HTTP and HTTPS requests from a particular client are always sent to the same Real Server as shown below:

SSL	192.168.110.50	Ports 80,443/tcp	Direct Routing	Add a new Real Server	
SSL1	192.168.110.51	80,443	Weight 100	Modify	Delete
SSL2	192.168.110.52	80,443	Weight 100	Modify	Delete

SSL Termination on the Load Balancer (SSL Offloading)

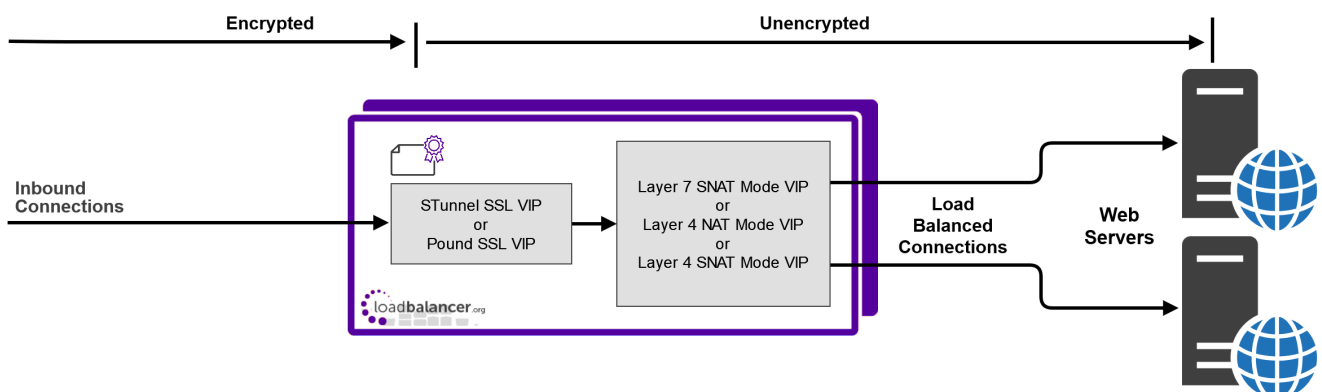
Note

SSL termination on the load balancer can be very CPU intensive. In most cases, for a scalable solution, terminating SSL on the Real Servers is the best option.

SSL Termination can be configured using STunnel, Pound and HAProxy.

Using STunnel or Pound to Terminate SSL

Here, either an STunnel or Pound SSL Virtual Service is used to decrypt the traffic. Once decrypted, traffic is passed to a second VIP which handles the load balancing between the backend servers as shown below.

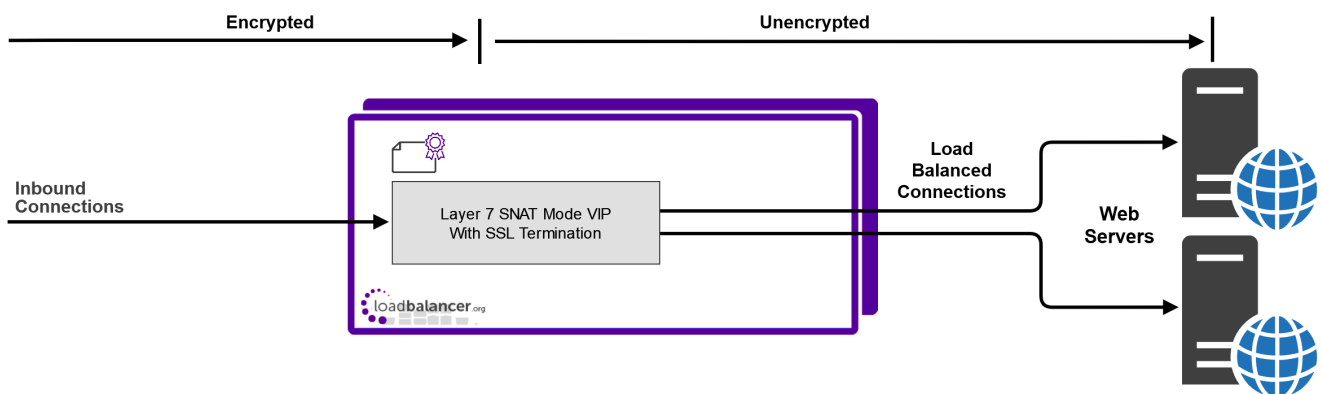


Notes

1. By default, a self-signed certificate is used for the new VIP. Certificates can be created or uploaded as explained in [Certificates](#).
2. The backend for the STunnel / Pound VIP can be a Layer 7 SNAT mode VIP, a Layer 4 NAT mode VIP or a layer 4 SNAT mode VIP.
3. If a layer 7 SNAT mode VIP is used as the backend for the STunnel or Pound VIP, cookie based persistence as well as all other layer 7 techniques can be used to control traffic flow to the Real Servers.

Using HAProxy to Terminate SSL

Here, a single layer 7 SNAT mode VIP handles both the decryption and the load balancing between the backend servers as shown below.



Notes

1. By default, a self-signed certificate is used for the new VIP. Certificates can be created or uploaded as explained in [Certificates](#).
2. Cookie based persistence as well as all other layer 7 techniques can be used to control traffic flow to the Real Servers.

Certificates

If you already have an SSL certificate in either PFX or PEM file format, this can be uploaded to the Load balancer using the certificate upload option as explained in [Uploading Certificates](#). Alternatively, you can create a Certificate Signing Request (CSR) and send this to your CA to create a new certificate, or you can create a locally signed custom certificate.

Generating a CSR on the Load Balancer

CSRs can be generated on the load balancer to apply for a certificate from your chosen CA.

To generate a CSR:

1. Using the WebUI, navigate to: *Cluster Configuration > SSL Certificates*.
2. Click **Add a new SSL Certificate** & select *Create a New SSL Certificate (CSR)*.

I would like to:

☐ Upload prepared PEM/PFX file

☒ Create a new SSL Certificate Signing Request (CSR) ?

☐ Create a new Self-Signed SSL Certificate.

Label ?

Domain (CN) ?

Subject Alternate Name ?

Organisation (O) ?

Organisation unit (OU) ?

City (L) ?

State or Province (ST) ?

Country code (C) ?

Email address ?

CSR Key Length ?

Create

=

3. Enter a suitable **Label** (name) for the certificate.
4. Populate the remaining fields according to your requirements.



Note

To specify multiple SANs, separate each name with a comma.

5. Once all fields are complete click **Create**.
6. To view the CSR click **Modify** next to the new certificate, then expand the Certificate Signing Request (CSR) section.
7. Copy the CSR and send this to your chosen CA.
8. Once received, copy/paste your signed certificate into the **Your Certificate** section.
9. Intermediate and root certificates can be copied/pasted into the **Intermediate Certificate** and **Root Certificate** sections as required.
10. Click **Update** to complete the process.

Generating a Self Signed Custom Certificate on the Load Balancer

To generate a self signed certificate:



1. Using the WebUI, navigate to: *Cluster Configuration > SSL Certificates*.
2. Click **Add a new SSL Certificate** & select *Create a new Self-Signed SSL Certificate*.
3. Enter a suitable *Label* (name) for the certificate.
4. Populate the remaining fields according to your requirements.

**Note**

To specify multiple SANs, separate each name with a comma.

5. Once all fields are complete click **Create**.

Uploading Certificates

Certificates in either PEM or PFX formats can be uploaded to the load balancer.

To upload a certificate:

1. Using the WebUI, navigate to: *Cluster Configuration > SSL Certificates*.
2. Click **Add a new SSL Certificate** & select *Upload prepared PEM/PFX file*.
3. Enter a suitable *Label* (name) for the certificate.
4. Browse to and select the certificate file to upload (PEM or PFX format).
5. Enter the password, if applicable.
6. Click **Upload Certificate**, if successful, a message similar to the following will be displayed:

Information: cert1 SSL Certificate uploaded successfully.

**Note**

If your Primary & Secondary are correctly configured as a clustered pair, when you upload the certificate file to the Primary, the file will be automatically copied over to the Secondary appliance.

**Note**

SSL certificates are included in the backup archive when a backup is created. For details, please refer to [Backup & Restore](#).

Checking where a Particular Certificate is Used

Certificates can be used for securing the WebUI, SSL termination (HAProxy, STunnel or Pound) as well as for client certificates with mTLS. A particular certificate can be used for multiple purposes and it's often useful to be able to quickly view where the certificate has been used.

To list where a particular certificate has been used:

1. Using the WebUI, navigate to: *Cluster Configuration > SSL Certificates*.
2. Move the mouse to hover over the "Used" text.



3. All current uses for the certificate will be listed as shown in the example below:

Search 1 Certificates	Show all				
Certificate Name	Current State	Size	Algorithm	Options	Delete
internal.lb.org	Certificate installed			SSL Info	Used.
					Used by: The WebUI. Stunnel SNI Rule: SSL-Web-Cluster/*.

Exporting PFX Certificates from Windows Servers

When exporting certificates from Windows servers, make sure that **Yes, export the private key** is selected, this will enable the output format to be PFX. Also make sure that **Include all certificates in the certification path if possible** is selected.

Creating a PEM file

Using a text editor such as vi or vim under Linux or Notepad under Windows, create an empty file (e.g. pem.txt) then copy/paste the entire contents of each of the following items into this file in the order listed below:

- Private Key
- SSL Certificate
- Intermediate Certificate
- Root CA Certificate

Make sure you include the beginning and end tags. The resulting file should look similar to the following:

```
-----BEGIN PRIVATE KEY-----
(the contents of your Private Key goes here)
-----END PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
(the contents of your SSL Certificate goes here)
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
(the contents of your Intermediate Certificate goes here)
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
(the contents of your Root Certificate goes here)
-----END CERTIFICATE-----
```

Once created, the file can be uploaded to the load balancer.

Converting between certificate formats

In some circumstances it may be required to manually convert certificates between formats. In these cases, OpenSSL can be used. OpenSSL is included on the appliance and is typically included by default in Linux distributions. For Windows, it can be downloaded [here](#). Once installed, you'll have an OpenSSL directory located on your filesystem (by default C:\OpenSSL). To use the program, open a command window, navigate to the location where it was installed (by default C:\OpenSSL\bin) then run the required command as shown below.

Converting PFX certificates to PEM format

```
openssl pkcs12 -in file.pfx -nodes -out file.pem
```

Converting .cer certificates to PEM format

```
openssl x509 -in file.cer -inform DER -out file.pem -outform PEM
```

Converting an Encrypted Private Key to an Unencrypted Key

If a password has been included in the private key, this should be removed before it is used with your PEM file. This can be done using the following command:

```
openssl rsa -in encrypted-server.key -out unencrypted-server.key
```

Let's Encrypt

Let's Encrypt is a zero cost Certificate Authority for HTTPS encryption, now trusted by all major root programs, including Google, Microsoft, Apple, Mozilla and Oracle. Used in conjunction with freely available tools it provides automatic enrollment/renewal, simple cert creation, negating validation emails and manual configuration.

For much more information, please refer to our following blogs:

- [Loadbalancer.org with Let's Encrypt - quick and dirty!](#)
- [Let's Encrypt – how did we survive without it?](#)
- [How to automate SSL/TLS certificate renewal with Let's Encrypt.](#)

Creating a SSL Termination

STunnel is used by default for all new SSL terminations. If you want to use Pound or HAProxy, you'll need to set the **SSL Operating Mode** to **Custom** as described in [SSL Operation Mode - Custom](#).

To add a SSL termination:

1. Using the WebUI, navigate to: **Cluster Configuration > SSL Termination**.
2. Click **Add a new Virtual Service**.



SSL Termination - Add a new Virtual Service

Label	<input type="text" value="SSL"/>	?
Associated Virtual Service	<input type="text" value="None"/>	?
Virtual Service IP Address	<input type="text"/>	
Virtual Service Port	<input type="text" value="443"/>	?
Backend IP Address	<input type="text"/>	
Backend Virtual Service Port	<input type="text" value="80"/>	?
SSL Operation Mode	<input type="text" value="High Security"/>	
SSL Certificate	<input type="text" value="Default Self Signed Certificate"/>	?
Source IP Address	<input type="text"/>	?
Enable Proxy Protocol	<input type="checkbox"/>	?
Bind Proxy Protocol to L7 VIP	<input type="text" value="None"/>	?

3. Set the *Associated Virtual Service* to the required value:

- If you leave it set to **None**:
 - Enter the required **Label** (name) for the Virtual Service.
 - Enter the required **Virtual Service IP Address**.
 - Enter the required **Virtual Service Port** - typically **443**.
 - Enter the required **Backend IP Address** - This is normally the same IP address as the Virtual Service IP address but can be any valid IP. The IP address specified must correspond to a Layer 7 SNAT mode VIP or a Layer 4 NAT / SNAT mode VIP. Unencrypted traffic will be sent here for load balancing.
 - Enter the required **backend Virtual Service Port** - typically **80**.
 - Select the required **SSL Operation Mode**:

Note

For more information see [SSL Operation Mode](#).

- **High Security** - Configure the STunnel VIP for high security
- **FIPS Compliant** - Configure the STunnel VIP for FIPS compliance
- **High Compatibility** - Configure the STunnel VIP for high compatibility
- **Custom** - All settings can be configured manually

**Note**

For more information see [SSL Operation Mode - Custom](#).

- Select the required **SSL Certificate**.
 - Set the required **Source IP Address** - by default the **Virtual Service IP Address** is used but this can be changed to any other address owned by the load balancer if required.
 - Configure **Enable Proxy Protocol** - If you wish to use HAProxy and the Proxy Protocol this option needs to be enabled (checked) to allow SSL termination on the load balancer whilst passing the client's IP address to the Real Servers. This option only enables a Proxy ACL Rule on a Single STunnel VIP.
 - Configure **Bind Proxy Protocol to L7 VIP** - If **Enable Proxy Protocol** is enabled, selecting a layer 7 Virtual service here configures the layer 7 service to expect the proxy protocol from this STunnel service. This enables the layer 7 service to pass the client's IP in a X-Forwarded-For header or with TProxy while still accepting HTTP traffic on the same port (for more information, please refer to [Transparency at Layer 7](#)). Note that manually defined layer 7 VIPs are not included in the dropdown.
- If you select a particular VIP where you want to forward the unencrypted STunnel traffic:
- Enter the required **Label** (name) for the Virtual Service.

**Note**

The label will be auto configured based on the **Associated Virtual Service** selected. This can be changed if required.

- Enter the required **Virtual Service Port** - typically **443**.
- Select the required **SSL Operation Mode**:

**Note**

For more information see [SSL Operation Mode](#).

- **High Security** - Configure the STunnel VIP for high security
- **FIPS Compliant** - Configure the STunnel VIP for FIPS compliance
- **High Compatibility** - Configure the STunnel VIP for high compatibility
- **Custom** - All settings can be configured manually




**Note**

For more information see [SSL Operation Mode - Custom](#).

- Select the required **SSL Certificate**.
- Click **Update** to create the new STunnel Virtual Service.

Once the SSL Termination is created, the associated VIP will be displayed with a padlock symbol in the system overview as shown below:



VIRTUAL SERVICE	IP	PORTS	CONNS	PROTOCOL	METHOD	MODE
  VIP1	192.168.111.232	80	0	HTTP	Layer 7	Proxy 

SSL Operation Mode

The following STunnel settings are auto-configured for each SSL Operation Mode:

STunnel Setting	High Security	FIPS Compliant	High Compatibility
Disable SSLv3 Ciphers	✓	✓	✓
Disable TLSv1.0 Ciphers	✓	✓	✗
Disable TLSv1.1 Ciphers	✓	✓	✗
Disable TLSv1.2 Ciphers	✗	✗	✗
Disable TLSv1.3 Ciphers	✗	✗	✗
Do Not Insert Empty Fragments	✓	✓	✓
Delay DNS Lookups	✓	✓	✓
Honor Cipher Order	✓	✓	✓
Disable SSL Renegotiation	✓	✓	✓

The following SSL Ciphers are auto-configured for each SSL Operation Mode:

High Security

ECDHE-ECDSA-AES256-GCM-SHA384 : ECDHE-ECDSA-AES128-GCM-SHA256 : DHE-RSA-AES256-GCM-SHA384 : DHE-RSA-AES128-GCM-SHA256 : ECDHE-RSA-AES256-GCM-SHA384 : ECDHE-RSA-AES128-GCM-SHA256

FIPS Compliant

ECDHE-RSA-AES256-GCM-SHA384 : ECDHE-ECDSA-AES256-GCM-SHA384 : ECDHE-RSA-AES256-SHA384 : ECDHE-ECDSA-AES256-SHA384 : DHE-DSS-AES256-GCM-SHA384 : DHE-RSA-AES256-GCM-SHA384 : DHE-RSA-AES256-SHA256 : DHE-DSS-AES256-SHA256 : AES256-GCM-SHA384 : AES256-SHA256 : ECDHE-RSA-AES128-GCM-SHA256 : ECDHE-ECDSA-AES128-GCM-SHA256 : ECDHE-RSA-AES128-SHA256 : ECDHE-ECDSA-AES128-SHA256 : DHE-DSS-AES128-GCM-SHA256 : DHE-RSA-AES128-GCM-SHA256 : DHE-RSA-AES128-SHA256 : DHE-DSS-AES128-SHA256 : AES128-GCM-SHA256 : AES128-SHA256 : AES256-SHA : AES128-SHA

High Compatibility

ECDHE-RSA-AES256-GCM-SHA384 : ECDHE-RSA-AES128-GCM-SHA256 : AES128-GCM-SHA256 : AES256-SHA256 : AES128-SHA256 : AES256-SHA : AES128-SHA : DHE-RSA-AES256-SHA256

Custom (Initial Setting)



ECDHE-RSA-AES256-GCM-SHA384 : ECDHE-RSA-AES128-GCM-SHA256 : AES128-GCM-SHA256 : AES256-SHA256 : AES128-SHA256 : AES256-SHA : AES128-SHA : DHE-RSA-AES256-SHA256

SSL Operation Mode - Custom

If you set the *SSL Operating Mode* to **Custom** the following SSL/TLS settings can be configured manually:

1. *Ciphers to use* - the default is:

ECDHE-RSA-AES256-GCM-SHA384 : ECDHE-RSA-AES128-GCM-SHA256 : AES128-GCM-SHA256 : AES256-SHA256 : AES128-SHA256 : AES256-SHA : AES128-SHA : DHE-RSA-AES256-SHA256

This can be modified as required, or the field can be cleared (blank) to allow all available ciphers (not recommended)

2. *Disable SSLv3 Ciphers* - When checked this option disables all SSLv3 Ciphers.
3. *Disable TLSv1.0 Ciphers* - When checked this option disables all TLSv1.0 Ciphers.
4. *Disable TLSv1.1 Ciphers* - When checked this option disables all TLSv1.1 Ciphers.
5. *Disable TLSv1.2 Ciphers* - When checked this option disables all TLSv1.2 Ciphers.
6. *Disable TLSv1.3 Ciphers* - When checked this option disables all TLSv1.3 Ciphers.
7. *SSL Terminator* - the options are **STunnel**, **HAProxy** or **Pound**.
 - If **STunnel** is selected:
 - *Do not Insert Empty Fragments* - This option should be enabled (checked) to ensure mitigation of both the BEAST and CRIME MITM attacks. It is also required for PCI Testing.
 - *Delay DNS Lookups* - This option is useful for dynamic DNS, or when DNS is not available during STunnel startup (road warrior VPN, dial-up configurations).
 - *Honor Cipher Order* - When choosing a cipher during an SSLv3 or TLSv1 handshake, normally the client's preference is used. If this directive is enabled, the server's preference will be used instead.
 - *Disable SSL Renegotiation* - Applications of the SSL renegotiation include some authentication scenarios, or re-keying long lasting connections. On the other hand this feature can facilitate a trivial CPU-exhaustion DoS attack. This option should be enabled (checked) to mitigate the BEAST Attack.
 - *Time to Close* - Configure the global client response timeout in seconds. This setting should not require changing.

- If **HAProxy** is selected:

Note

To use HAProxy for SSL termination, the *Associated Virtual Service* field must be set to a layer 7 VIP.

Note

It's also possible to create an HAProxy SSL termination at the same time that a layer 7 VIP is created. When creating the VIP, click **[Advanced]** and enable (check) **Create**

- **CA Certificate** - If you want to use client certificate authentication, set the relevant CA Certificate here.

 **Note**

To configure a **CA Certificate**, create a PEM file with the complete CA certificate chain and private key, then use the WebUI menu option: **Cluster Configuration > CA Certificate Families** to configure a new CA Certificate Family and upload the PEM file. Multiple certificates can be associated with each family, the first certificate is activated by default. For more information about mTLS, please refer to [Mutual Transport Layer Security \(mTLS\)](#).

- **CA Certificate Verification** - Select the required certificate verification level.
 - **Required** - Connections from clients with invalid certificates will be dropped immediately.
 - **Optional** - Connections with an invalid certificate will be allowed to the associated Virtual Service where further actions may be taken using ACL rules. For example, redirecting clients with an invalid certificate to a custom error page.
 - **Never** - No certificate verification will occur. For example, this could be useful if the certificate was signed by a private certificate authority.
- **ALPN** - Select one or more (using the CTRL key) Application-Layer Protocol Negotiation protocol IDs to advertise for this termination.
- If **Pound** is selected:
 - **Enable WebDAV Verbs** - Selecting this option permits the use of the following commands:
 - Extended HTTP Requests: PUT, DELETE
 - Standard WebDAV verbs: LOCK, UNLOCK, PROPFIND, PROPPATCH, SEARCH, MKCOL, MOVE, COPY, OPTIONS, TRACE, MKACTIVITY, CHECKOUT, MERGE, REPORT
 - Microsoft WebDAV extensions: SUBSCRIBE, BPROPPATCH, POLL, BMOVE, BCOPY, BDELETE, CONNECT
 - **Header Field Name & Header Field Value** - Use to configure a custom Pound header.
 - **Rewrite HTTP Redirects** - If they point to the backend itself or to the listener (but with the wrong protocol) the response will be changed to show the virtual host in the request.
 - **Honor Cipher Order** - When choosing a cipher during an SSLv3 or TLSv1 handshake, normally the client's preference is used. If this directive is enabled, the server's preference will be used instead. This option should be enabled to mitigate the BEAST attack.
 - **Client Cipher Renegotiation** - Sets whether the client is allowed to renegotiate the cipher order. This option should be set to "No Client Renegotiation" to mitigate the BEAST attack. The options are:
 - No Client Renegotiation - no client renegotiation will be honored
 - Secure Renegotiation - secure renegotiation will be honored

- Insecure Renegotiation - insecure renegotiation will be honored

Server Name Indication (SNI)

Server Name Indication (SNI) rules can be configured for STunnel VIPs. SNI is an extension to the TLS protocol which allows a client to indicate which hostname it is attempting to connect to at the start of the handshaking process. This allows the load balancer to present the correct certificate based on the FQDN requested and route requests to the appropriate layer 7 VIP or a custom IP address / port.

Configuring Server Name Indication (SNI) Rules

SNI rules cannot be added at the same time the STunnel VIP is initially created. They must be added afterwards using the modify option.

To configure SNI rules:

1. Using the WebUI, navigate to: *Cluster Configuration > SSL Termination*.
2. Click **Modify** next to the relevant STunnel VIP.
3. Scroll to the bottom of the screen and click **New SNI Rule**.

Friendly Name	<input type="text" value="rule1"/>
SNI to match	<input type="text" value="www.loadbalancer.org"/>
SSL Certificate	Default Self Signed Certificate ▾
Associated Virtual Service	VIP1 ▾

Add Rule

4. Enter an appropriate *Friendly Name* for the new rule, e.g. **rule1**.
5. Enter the required *SNI to Match* e.g. **www.loadbalancer.org**.



Note

Wildcard entries such as ***.loadbalancer.org** are also supported. Make sure that the SSL certificates match accordingly.

6. Select the required *SSL Certificate*.
7. Set the *Associated Virtual Service* dropdown according to your requirements, either:
 - Select the required Virtual Service where you'd like to forward traffic.
 - Or select **Custom**, specify the *Backend IP Address* and *Backend Virtual Service Port* and configure *Enable Proxy Protocol* as needed - Proxy Protocol is enabled by default.
8. Click **Add Rule**.
9. Repeat the above steps to add additional rules.
10. Once the rules are added, they're displayed in a list under the *Current SNI Rules* section as shown in the example below:



Current SNI Rules

Search SNI Rules						Delete
SNI Name	SNI to Match	Certificate	Service	Proxy Protocol	Modify	Delete
rule1	www.loadbalancer.org	server	VIP1	<input checked="" type="checkbox"/>	Modify	<input type="checkbox"/>
rule2	www.lptestdomain.com	server	VIP1	<input checked="" type="checkbox"/>	Modify	<input type="checkbox"/>

Delete

11. Modify or delete SNI rules using the buttons provided.
12. To apply the new settings, restart STunnel using the **Reload STunnel** button in the "Commit changes" box at the top of the screen.
13. Once SNI rules have been configured for a particular STunnel VIP, this is indicated next to the STunnel VIP name as shown below:

Service Name	IP & Port	Backend & Port	Options
 SSL	192.168.110.235:443	192.168.110.235:80	Modify SSL Info Delete

SSL Termination on the Load Balancer with Re-encryption (SSL Bridging)

Note

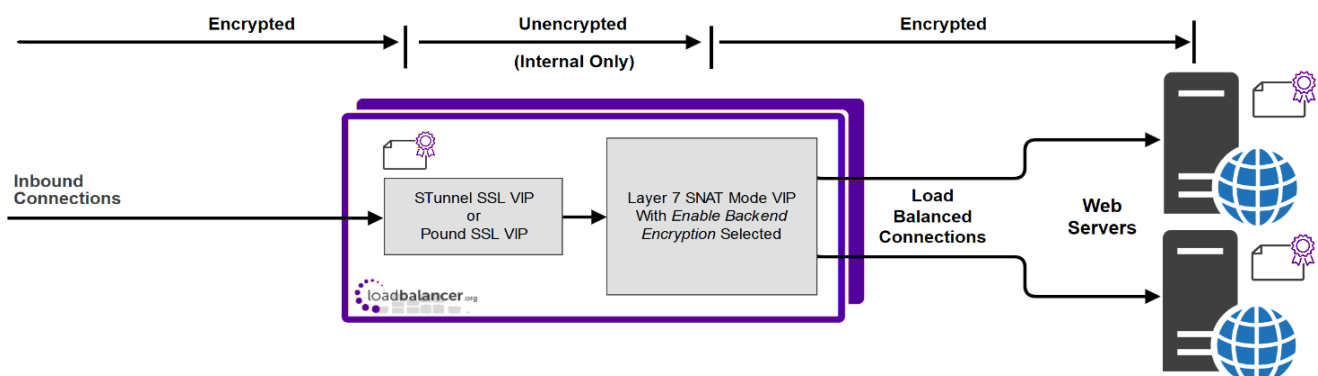
SSL termination on the load balancer can be very CPU intensive. In most cases, for a scalable solution, terminating SSL on the Real Servers is the best option.

Note

For information on configuring SSL termination, please refer to [SSL Termination on the Load Balancer \(SSL Offloading\)](#).

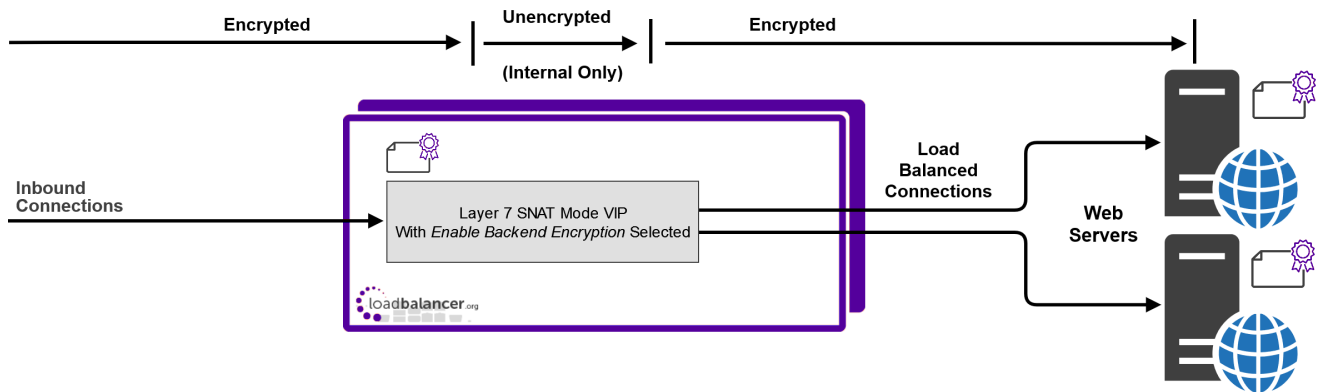
Using STunnel or Pound to Terminate SSL

Here, SSL is terminated using STunnel or Pound. Un-encrypted traffic and is then passed to a layer 7 VIP which handles the re-encryption as shown below.



Using HAProxy to Terminate SSL

Here, SSL termination and re-encryption are both handled by a single layer 7 VIP as shown below.



Enabling SSL Re-Encryption

Backend encryption can be enabled at the Virtual Service level which then applies to all associated Real Servers or for each Real Server as detailed below.

To enable re-encryption at the Virtual Service level:

1. Using the WebUI, navigate to: **Cluster Configuration > Layer 7 - Virtual Services**.
2. Click **Modify** next to relevant Virtual Service.
3. Scroll down to the **SSL** section and enable (check) the **Enable Backend Encryption** checkbox. If there are existing Real Servers, you'll be asked if you want to apply the new setting to those servers - click **OK** or **Cancel** as required.

SSL		[Advanced +]
Enable Backend Encryption	<input checked="" type="checkbox"/>	?
Other		[Advanced +]
		Cancel Update

4. The following default ciphers are used:

```
ECDHE-ECDSA-AES256-GCM-SHA384 : ECDHE-ECDSA-AES128-GCM-SHA256 : DHE-RSA-AES256-GCM-SHA384 : DHE-RSA-AES128-GCM-SHA256 : ECDHE-RSA-AES256-GCM-SHA384 : ECDHE-RSA-AES128-GCM-SHA256
```

To change this, click **[Advanced]** in the **SSL** section, enable (check) the **Use Custom Ciphers** checkbox and specify the required cipher in the **Default Real Server Ciphers** field.

5. Click **Update**.

To enable re-encryption at the Real Server level:

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Real Servers*.
2. Click **Modify** next to relevant Real Server.

Layer 7 Add a new Real Server

Label	<input type="text" value="IIS1"/>	?
Real Server IP Address	<input type="text" value="192.168.210.240"/>	?
Real Server Port	<input type="text" value="443"/>	?
Re-Encrypt to Backend	<input checked="" type="checkbox"/>	?
Enable Redirect	<input type="checkbox"/>	?
Weight	<input type="text" value="100"/>	?

Cancel Update

3. Enable (check) the option *Re-Encrypt to Backend*.
4. Click **Update**.

SSL - Advanced Configuration

Pound Global Settings

Lock Pound Configuration - When enabled it will stop the user interface overwriting the configuration files so manual changes can be made.

Logging - Activate detailed logging of the Pound SSL termination service. When activated the Pound log is written to /var/log/poundssl.log.

Client Timeout - Configure the global client response timeout in seconds. This setting should not require changing. The default is 30 seconds.

Global Server Timeout - Configure the global Real Server response timeout in seconds. This setting should not require changing.

Ulimit - This setting will change the maximum number of file descriptors available to the pound process. The default is 81000.

Process Threads - Start the Pound process with X number of threads. Note that these threads are allocated at start so if you're not using them they will take up memory needlessly. The default is 250.

Transparent Proxy - Enable TProxy support in Pound SSL. The combination of Pound, TProxy, and HAProxy allows SSL termination on the load balancer whilst passing the client's IP address to the Real Servers. This option also automatically enables TProxy for HAProxy.

Note

A consequence of using Transparent Proxy with both Pound and HAProxy is that you can no longer access the HAProxy Virtual Service directly. With transparency turned on, HAProxy will



only accept traffic from Pound. One way to get around this is to configure the HAProxy VIP to listen on two ports. One will listen on port 80, and be your standard HTTP service. The other will listen on a different port - 81 for example, and will be the destination for traffic from Pound. For more information, please refer to [Transparency at Layer 7](#).

STunnel Global Settings

STunnel Global Settings		
Debug Level	Emergency (0) ▾	?
Disable Nagle Algorithm	<input type="checkbox"/>	?
Enable FIPS 140-2 mode	<input type="checkbox"/>	?
		<button>Update</button>

Debug Level - Option to set the debugging level for all STunnel Services. The Debug Level is one of the syslog level names or numbers emergency (0), Alert (1), Critical (2), err (3), Warning (4), Notice (5), Information (6), or Debug (7). The higher the number the more detail will be contained in the STunnel Logs.

Disable Nagle Algorithm - With this option ticked (enabled) the Nagle Algorithm will be disabled. More details can be found in RFC 896.

Enable FIPS 140-2 Mode - FIPS (Federal Information Processing Standards) are a set of standards that describe document processing, encryption algorithms and other information technology standards for use within non-military government agencies and by government contractors and vendors who work with the agencies. Check to enable FIPS 140-2 mode for STunnel.

Mutual Transport Layer Security (mTLS)

mTLS is a way to establish a secure and encrypted connection between two entities. With mTLS, both the client and server have digital certificates that they use to verify their identities to each other before exchanging any data. This helps to ensure that the connection is secure and that the data being transmitted between the client and server cannot be intercepted or tampered with by any unauthorized parties.

Configuring mTLS

mTLS can be enabled on the front-end between the client and the load balancer and on the back-end between the load balancer and the load balanced servers.

The configuration steps below assume that there is an existing layer 7 VIP named **AppServers** with associated Real Servers **AppServer1**, **AppServer2** and **AppServer3**.

Upload the CA Certificate

In the example presented here, the same CA certificate is used to validate the connecting clients and also the backend servers. If different CAs are used, multiple CA Certificate Families can be configured.

1. Using the WebUI, navigate to: *Cluster Configuration > CA Certificate Families*.
2. Click **Create Family**.



Certificate family details		
Family label	<input type="text" value="mTLS"/>	?
Certificate label	<input type="text" value="ca-cert"/>	?
Certificate contents	<input type="button" value="Choose File"/> <input type="text" value="ca-cert.pem"/>	?

- Specify a suitable *Family label*, e.g. **mTLS**.
- Specify a suitable *Certificate Label*, e.g. **ca-cert**.
- Click **Choose File** and browse to and select the relevant PEM or PFX file.
- Click **Create**.

Front-end mTLS

Step 1 - Upload the Certificate to be Used for the SSL Termination

- Using the WebUI, navigate to *Cluster Configuration > SSL Certificate*.
- Click **Add a new SSL Certificate**.
- Select the **Upload prepared PEM/PFX file** option.

I would like to:

☒ Upload prepared PEM/PFX file
 ☐ Create a new SSL Certificate Signing Request (CSR)
 ☐ Create a new Self-Signed SSL Certificate.

Label

File to upload

- Specify an appropriate *label* (name), e.g. **server-cert**.
- Click **Choose File** and select the relevant PEM or PFX file.
- Click **Upload Certificate**.

Step 2 - Configure the SSL Termination and Enable mTLS

- Using the WebUI, navigate to *Cluster Configuration > SSL Termination* and click **Add a new Virtual Service**.

Label	SSL-AppServers	?
Associated Virtual Service	AppServers	?
Virtual Service Port	443	?
SSL Operation Mode	Custom	
SSL Certificate	server-cert	?
Ciphers to use	ECDHE-ECDSA-AES256-GCM-SHA384	?
Disable SSLv3 Ciphers	<input checked="" type="checkbox"/>	?
Disable TLSv1.0 Ciphers	<input checked="" type="checkbox"/>	?
Disable TLSv1.1 Ciphers	<input checked="" type="checkbox"/>	?
Disable TLSv1.2 Ciphers	<input type="checkbox"/>	?
Disable TLSv1.3 Ciphers	<input type="checkbox"/>	?
SSL Terminator	<input type="radio"/> STunnel <input checked="" type="radio"/> HAProxy <input type="radio"/> Pound	?
CA Certificate	mTLS	?
ALPN	<div> xmpp-server xmpp-client webRTC tds/8.0 sunrpc stun.turn stun.nat-discovery </div>	?

Cancel
Update

- Using the *Associated Virtual Service* dropdown, select the Virtual Service where mTLS should be enabled, e.g. **AppServers**.

Note

Once the VIP is selected, the *Label* field will be auto-populated with **SSL-AppServers**. This can be changed if preferred.

- Set the *SSL Operation Mode* to **Custom**.
- Set the *SSL Terminator* to **HAProxy**.
- Set the *SSL Certificate* to the SSL Certificate uploaded previously, e.g. **server-cert**.
- Set the *CA Certificate* to the CA Certificate Family created previously, e.g. **mTLS**.
- Click **Update** to create the SSL Termination.

Back-end mTLS

Step 1 - Upload the Certificate that will be Used to Prove the Identity of the Load Balancer

- Using the WebUI, navigate to *Cluster Configuration > SSL Certificate*.

2. Click **Add a new SSL Certificate**.
3. Select the **Upload prepared PEM/PFX file** option.

I would like to:

☒ Upload prepared PEM/PFX file

☐ Create a new SSL Certificate Signing Request (CSR)

☐ Create a new Self-Signed SSL Certificate.

Label:

File to upload:

4. Specify an appropriate *label* (name), e.g. **client-cert**.
5. Click **Choose File** and select the relevant PEM or PFX file.
6. Click **Upload Certificate**.

Step 2 - Enable mTLS

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 – Real Servers* and click **Modify** next to the first Real Server.
2. Configure the following Settings:

Label:

Real Server IP Address:

Real Server Port:

Re-Encrypt to Backend: ☒

Verify Server Certificate:

Send Client Certificate:

Enable Redirect: ☐

Weight:

Minimum Connections:

Maximum Connections:

- If desired, update the *Label* to indicate that mTLS is now enabled, e.g. **mTLS-AppServer1**.

- Update the *Real Server Port* field to the required port, e.g. **443**.
- Enable (check) the *Re-Encrypt to Backend* checkbox.
- Set the *Verify Server Certificate* dropdown to the certificate family created previously, e.g. **mTLS**.
- Set the *Send Client Certificate* dropdown to the certificate uploaded above, e.g. **client-cert**.

3. Click **Update**.

4. Repeat these steps for all remaining Real Servers.

Certificate Revocation List (CRL)

If required, a CRL can be uploaded and applied to the CA Certificate Family as described below:

1. Using the WebUI, navigate to: *Cluster Configuration > CA Certificate Families*.
2. Click **Modify** next to the CA Certificate Family, e.g. **mTLS**.
3. Click **Upload Certificate Revocation List**.
4. Click **Choose File** and browse to and select the required CRL file.
5. Click **Upload**.

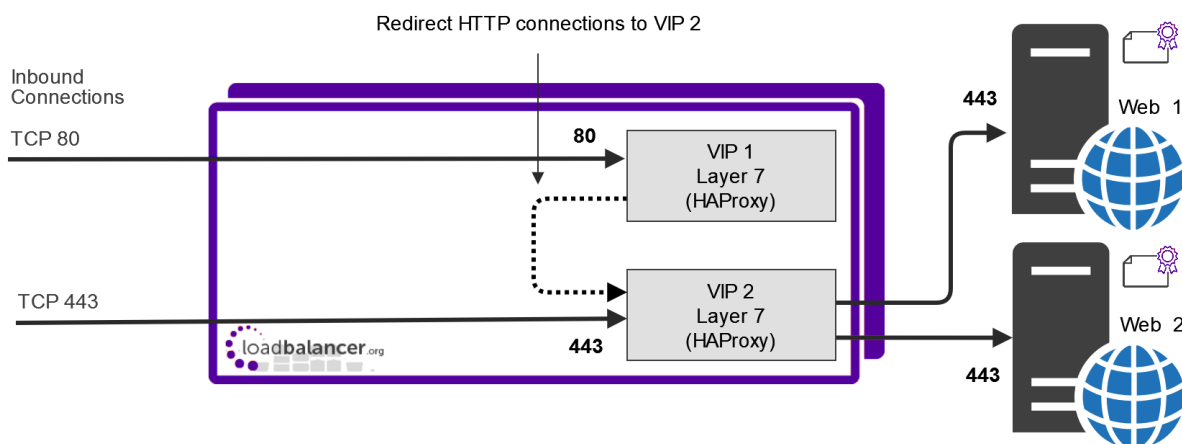
Once everything is configured, reload HAProxy using the **Reload HAProxy** button in the "Commit changes" box at the top of the screen to apply the new settings.

HTTP to HTTPS Redirection

HTTP to HTTPS redirection is supported both when terminating SSL on the Real Servers and when offloading SSL on the load balancer as described in the following sections.

When Terminating SSL on the Real Servers

This method requires two VIPs:



- **VIP 1** - This is a layer 7 HTTP mode VIP that listens on port 80 and redirects all requests to VIP 2.
 - This VIP does not require any Real Servers. Once configured, it will be shown purple/green in the System Overview.

- Force to HTTPS is enabled - this is set by modifying the VIP, scrolling to the **Other** section and enabling *Force to HTTPS*.

Note

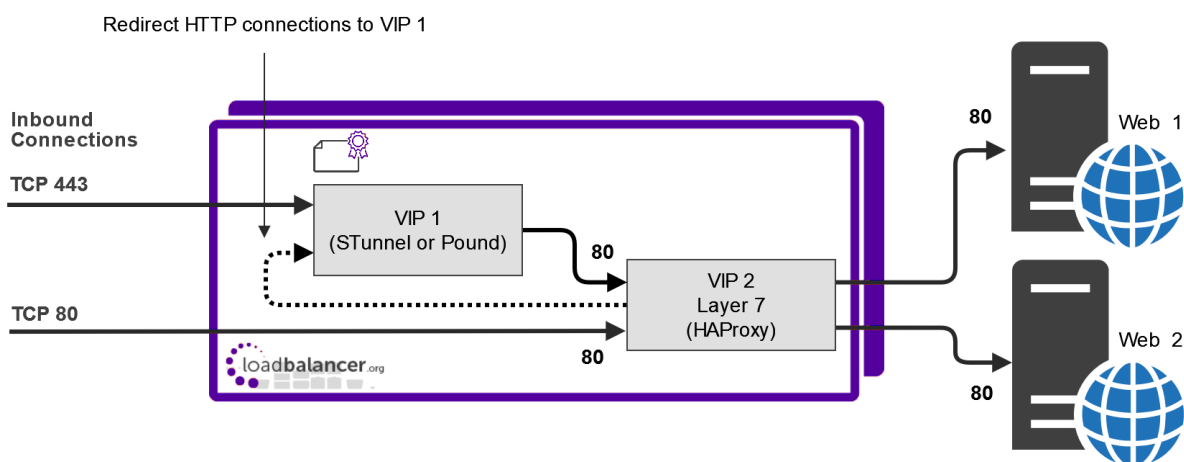
Force to HTTPS is only available when the VIP is in HTTP mode.

- **VIP 2** - This is a layer 7 TCP mode VIP that listens on port 443 and load balances connections between Web 1 & Web 2.
 - This VIP is configured on the same IP address as VIP 1.

When Terminating SSL on the Load Balancer

Using STunnel or Pound

This method requires two VIPs:



- **VIP 1** - This is a Pound or STunnel VIP that listens on port 443, terminates the SSL connection and then forwards the decrypted HTTP traffic to VIP 2 on port 80.
- **VIP 2** - This is a layer 7 HTTP mode VIP that listens on port 80 and load balances connections between Web 1 & Web 2.
 - This VIP is configured on the same IP address as VIP 1.
 - Force to HTTPS is enabled - this is set by modifying the VIP, scrolling to the **Other** section and enabling *Force to HTTPS*.

Note

Force to HTTPS is only available when the VIP is in HTTP mode.

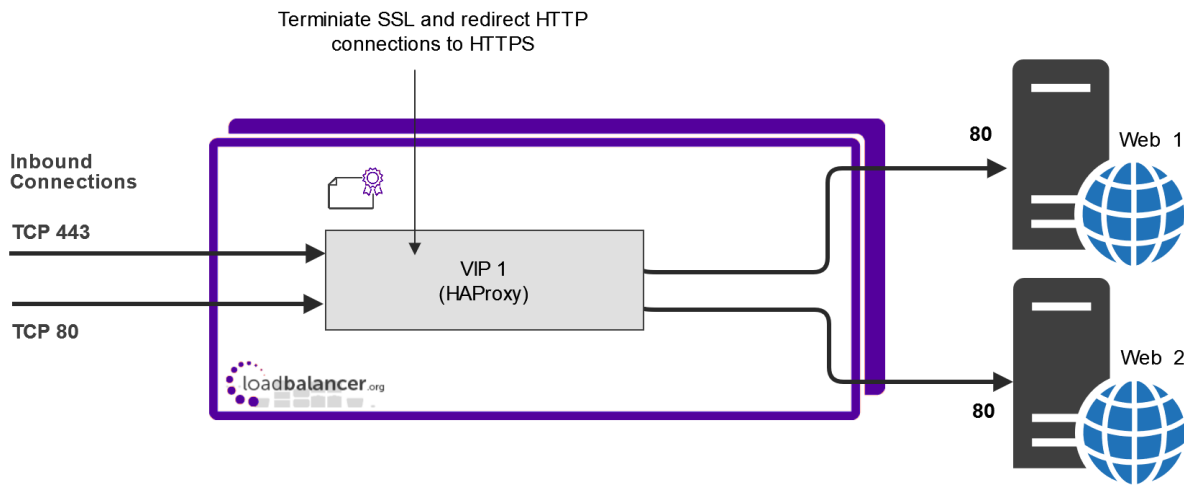
Note

If you want to re-encrypt the data from the load balancer to the Real Server, enable the **Re-encrypt to Backend** option for the each Real Server. For more information on using this option, please refer to [SSL Termination on the Load Balancer with Re-encryption \(SSL Bridging\)](#).

Using HAProxy

This method requires one VIP:





- **VIP 1** - This is a Layer 7 SNAT mode VIP configured in HTTP mode. It listens on port 80 and load balances the decrypted HTTP requests to Web 1 & Web 2.
 - This VIP also handles SSL termination. To configure SSL termination at the same time as creating the layer 7 VIP, click **[Advanced]** and in the **termination** section enable (check) the **Create HAProxy SSL Termination** option.
 - Force to HTTPS is enabled - this is set by modifying the VIP, scrolling to the **Other** section and enabling **Force to HTTPS**.

Note *Force to HTTPS* is only available when the VIP is in HTTP mode.

Note If you want to re-encrypt the data from the load balancer to the Real Server, enable the **Re-encrypt to Backend** option for the each Real Server. For more information on using this option, please refer to [SSL Termination on the Load Balancer with Re-encryption \(SSL Bridging\)](#).

Server Feedback Agent

The load balancer can dynamically modify the weight of each Real Server by gathering data from either a custom feedback agent or a HTTP server. Reducing the weight of a server compared to others in the pool will reduce the amount of traffic it receives.

For layer 4 VIPs, both the agent and HTTP Server methods can be used, for Layer 7 VIPs, only the agent method is supported.

By default, the agent listens on TCP port 3333, although this can be changed if required.

A telnet to port 3333 on a Real Server with the agent installed returns the current idle value as an integer value between 0 and 100. By default, the idle value is based on current CPU utilization. This can also be based on RAM utilization and the number of current connections or a combination of all three.

This can be configured by modifying the XML configuration file located in the agent's installation folder - by default C:\ProgramData\LoadBalancer.org\LoadBalancer. The file can be edited directly or by clicking the **Configuration** button in the agent monitor program - see "Controlling the Agent" below.

The load balancer uses the formula **(idle value/100) * initial weight** to find the new dynamic weight.

Note

The "initial weight" is the weight set in the WebUI for each Real Server.

For more information about the feedback agent, please refer to [this blog](#).

Windows Agent

The latest Windows feedback agent (v4.6.0) can be downloaded [here](#).

Installing the Agent

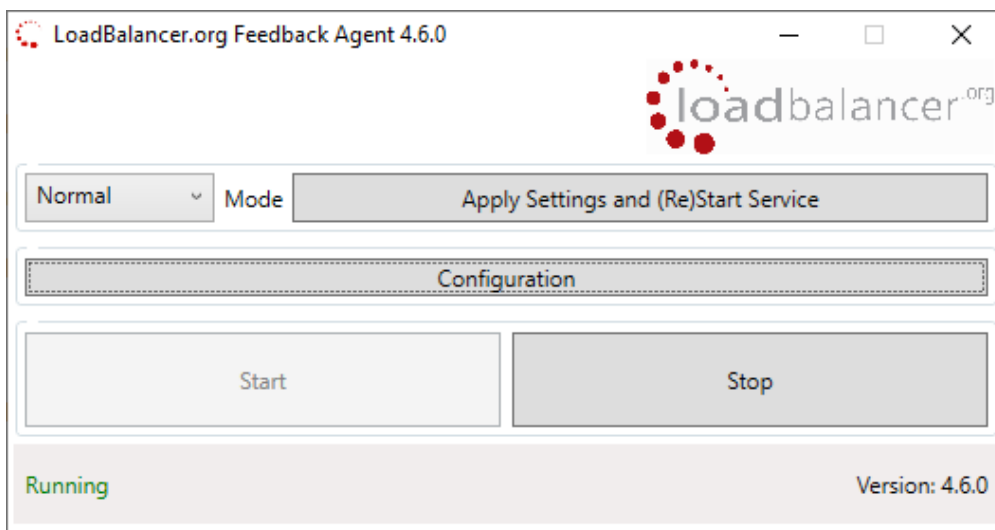
To install the agent, run **loadbalanceragent.msi**. Once the installation is complete, the Feedback Agent service is started automatically.

Note

The agent must be installed on each Real Server.

Controlling the Agent

The Feedback Agent service can be controlled and configured using the *Loadbalancer.org Feedback Agent* monitor program. By default this can be accessed from: **Windows Start Menu > Loadbalancer.org**.



Linux/Unix Agent

The Linux feedback agent files can be downloaded using the following links:

readme file: <https://downloads.loadbalancer.org/agent/linux/v4.1/readme.txt>

xinetd file: <https://downloads.loadbalancer.org/agent/linux/v4.1/lb-feedback>

feedback script: <https://downloads.loadbalancer.org/agent/linux/v4.1/lb-feedback.sh>

Installation & Testing

Install xinetd - if not already installed:

```
apt-get install xinetd
```



Insert the following line into /etc/services:

```
lb-feedback 3333/tcp # Loadbalancer.org feedback daemon
```

Then run the following commands:

```
cp lb-feedback.sh /usr/bin/lb-feedback.sh
chmod +x /usr/bin/lb-feedback.sh
cp lb-feedback /etc/xinetd.d/lb-feedback
chmod 644 /etc/xinetd.d/lb-feedback
/etc/init.d/xinetd restart
```

To test the agent:

```
telnet 127.0.0.1 3333
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
95%
Connection closed by foreign host.
```



Note

The agent files must be installed on each Real Server.

HTTP Server

You can use any HTTP server responding on port 3333 to give feedback information to the load balancer. The format of this information must be an integer number of 0-100 without any header information. Using this method, you can generate a custom response based on your application's requirements.

Configuring VIPs To Use The Agent or HTTP Server

To Configure Virtual Services to use Agent or HTTP Server (layer 4 only) feedback:

1. Using the WebUI, navigate to *Cluster Configuration > Layer 4 - Virtual Services* or *Cluster Configuration > Layer 7 - Virtual Services*.
2. Click **Modify** next to the relevant Virtual Service.

Feedback Method

Feedback Method

Agent ▼



Feedback Agent Port

3333



- For layer 4 VIPs set the Feedback Method to either **Agent** or **HTTP** depending on your requirements.
- For layer 7 VIPs set the Feedback Method to **Agent**.

3. Click **Update**.



4. Reload/restart services as prompted.

Global Server Load Balancing (GSLB)

GSLB enables traffic to be load balanced across multiple geographically dispersed servers. When used in conjunction with the failover and high availability features of the appliance, GSLB enables highly available, multi-site load balanced environments to be created. GSLB functionality is provided using the Open Source [Polaris GSLB](#).

Key Concepts

GSLB enables the load balancer(s) to provide intelligent DNS responses to inbound client queries for one or more sub domains. The responses given depend on the health of each endpoint and if Topology is configured, the location of those endpoints relative to the client making the request. Where GSLB is deployed along side application load balancing, the endpoints are usually the VIPs that are configured at each site. Where application load balancing is not used and only GSLB is configured, the endpoints are normally the Real Servers.

DNS delegation is used to delegate responsibility for the sub domain(s) to the GSLB service on the load balancers. Once delegated, it is the GSLB service on the load balancers that is responsible for providing the response to DNS queries for that sub domain.

In a two site setup with an HA pair of load balancers in each site, once GSLB and DNS delegation are correctly configured, the four load balancers act as intelligent name servers for the sub domains specified.

Key features

- Reliable health checking service supporting both TCP, HTTP(S) and custom external checks so that only healthy members/endpoints are returned on lookups
- Failover, round robin and also a topology method that directs clients to servers in the same location
- Can return single or multiple (up to 1024) answers at once
- Option to fallback to any healthy server or refuse the query
- Uses EDNS Client Subnet (ECS) by default so the client IP address / subnet is used when considering topology. For more details, please refer to [GSLB Advanced Configuration](#).

Note

For additional information, please refer to our [GSLB blog](#).

GSLB Configuration

GSLB is configured using the WebUI menu option: **Cluster Configuration > GSLB Configuration**. Four tabs are used to configure GSLB as illustrated below:

GSLB Configuration

Global Names	Members	Pools	Topologies
<div>New Global Name</div> <div>No Data</div>			

Global Names - Define the sub domain(s) that GSLB responds to.

Members - Are returned to clients in DNS responses, also known as "endpoints".

Pools - Associate one or more Global Names with the relevant Members. Also defines health checks, load balancing method, timeouts and various other settings.




Topologies - Define how network subnets and host addresses map to sites. In a multi-site deployment, this can be used to ensure that clients connect to the nearest / most appropriate site. A topology groups together the member(s) for a particular site and the clients who would normally connect via those members. Members and clients can be specified using a subnet address such as 10.0.0.0/24 or a host address such as 10.0.0.1/32.

The following table details the options in each tab.

Tab	Setting	Description
Global Names	<i>Name</i>	Name can be a combination of "0-9", "a-z", "A-Z", "-" (dash), "_" (underscore) or a "." (dot).
	<i>Hostname</i>	A valid RFC 1123 hostname, for example www.example.com.
	<i>TTL</i>	TTL is how long to cache the (hostname) DNS response in seconds. For example 3600 would be equal to 1 hour, the minimum (and default) value is "0" (0s).
Members	<i>Name</i>	Name can be a combination of "0-9", "a-z", "A-Z", "-" (dash), "_" (underscore) or a "." (dot).
	<i>IP</i>	A valid IPv4 address, for example 10.0.1.1.
	<i>Monitor IP</i>	A valid IPv4 address, for example 10.0.1.1.
	<i>Weight</i>	Weight of the server, min: 0 (server is disabled), max: 10.
Pools	<i>Name</i>	Name can be a combination of "0-9", "a-z", "A-Z", "-" (dash), "_" (underscore) or a "." (dot).
	<i>Monitor</i>	The type of health check to use. The options are:

Tab	Setting	Description
	Monitor - HTTP	<p>Perform a HTTP or HTTPS GET depending on Monitor Use SSL. Succeeds if HTTP response status is 200 or one of the codes specified in Monitor Expected codes.</p> <ul style="list-style-type: none"> • Monitor Use SSL - Whether to use SSL, default is "No" (false). • Monitor Hostname - Hostname to supply in HTTP Host: header. When using SSL this will also be supplied in SNI. • Monitor URL Path - A URL path to request, appended after the member's IP address, default is "/". • Monitor Port - Which port to check. If a value is not provided, port 80 will be used when Monitor Use SSL is false, port 443 will be used when Monitor Use SSL is true. • Monitor Expected Codes - A comma separated list of HTTP codes to match in a response. Valid range is between 100 and 599.
	Monitor - TCP	<p>Perform a TCP connect. Optionally: send text, read response and match a regex pattern.</p> <ul style="list-style-type: none"> • Monitor Port - Which port to check. • Monitor Send String - A string to send after connecting to the port, for example "check". • Monitor Match Return - A regex pattern to match (not case sensitive) in the response, for example "up".
	Monitor - Forced	<p>Forces a member to be either UP or DOWN effectively disabling health checking.</p> <ul style="list-style-type: none"> • Monitor Status - A string, either "up" or "down", default is "up".

Tab	Setting	Description
	<i>Monitor - External</i>	<p>Will run the script selected in the Monitor Script dropdown. The check will receive the IP address and port of the member and any additional arguments that are passed. If a Monitor Result is set, the check will be deemed a success if the script returns the configured string. If there is no Monitor Result the exit code will be used.</p> <ul style="list-style-type: none"> • Monitor Port - see above. • Monitor Script - Specify the script to use. • Monitor Parameters - A single or double quoted, comma separated list of additional parameters you may wish to pass in. This value is not required. <div> <p>The parameters that are passed into the shell script are as follows:</p> <ul style="list-style-type: none"> ▪ \$1 = pool member IP address ▪ \$2 = monitor port ▪ \$3 and onwards are taken from Monitor Parameters </div> <ul style="list-style-type: none"> • Monitor Result - A string to match in the response, for example "success". <div> <p>For more details on adding additional health check scripts, please refer to External Health Check Scripts (GSLB).</p> </div>

Tab	Setting	Description
	<i>Monitor - External Dynamic Weight</i>	<p>This will dynamically adjust the weight based on the output of the health check script. It should output between 0 and 10, 10 being of the highest priority and 0 being offline and removed from the pool. The exit code should be 0 at all times, anything else will report as a health check failure.</p> <ul style="list-style-type: none"> • Monitor Port - see above. • Monitor Script - see above. • Monitor Parameters - see above. <div>  Note <p>The "Feedback_Agent_Weight" GSLB template can be used to create a dynamic weight health check script that interacts with the Loadbalancer.org Feedback Agent.</p> </div> <div>  Note <p>For more details on adding additional health check scripts, please refer to External Health Check Scripts (GSLB).</p> </div>
	<i>LB Method</i>	<p>The load balancing method to use. The options are:</p> <ul style="list-style-type: none"> • wrr - Weighted round robin. Round robin with weighting. • twrr - Topology weighted round robin. As above but the topology file is also considered to direct clients to end-points in the same region/data center. • fogroup - Failover group. With this method, the first healthy end-point is handed out continuously unless it becomes unhealthy, then, the next healthy end-point is used, etc. Can be used for active-backup scenarios.
	<i>Global Names</i>	<p>A Pool can be associated with one or more global names. A pool requires at least one global name. Use the CTRL key to select multiple global names.</p>
	<i>Members</i>	<p>A pool must have at least one endpoint member. Drag and drop the endpoints from Available Members to _Members in Use.</p> <div>  Note <p>If the load balancing method is set to fogroup, the order in which the members are listed should be the same as the order in which they are intended to be used.</p> </div>
	Advanced > <i>Monitor Interval</i>	<p>In seconds, min: 1, max: 3600.</p>

Tab	Setting	Description
	Advanced > <i>Monitor Timeout</i>	In milliseconds, min: 100 (0.1s), max: 10000 (10 seconds).
	Advanced > <i>Monitor Retries</i>	Retry min: 0, max: 5, default is "0" no retries.
	Advanced > <i>Fallback</i>	Resolution behavior when all members of the pool are DOWN. The options are: <ul style="list-style-type: none"> • any - Perform distribution among all the configured members with non-zero weight ignoring the health status. This is the default. • refuse - Refuse all queries. Note: fallback is set to "any" with all member weights set to 0 will result in a NOERROR response with no answer section data.
	Advanced > <i>Max Addresses Returned</i>	Maximum number of A records to return in response, large responses will go over TCP min: "1", max: "1024", default: "1".
Topologies	<i>Name</i>	Name can be a combination of "0-9", "a-z", "A-Z", "-" (dash), "_" (underscore) or a "." (dot).
	<i>IP/CIDR</i>	A comma separated list of valid IPv4 address or CIDR denoted IP blocks for example "10.1.1.0/24, 10.1.1.2".

GSLB Service IP Address & Port

By default, the GSLB service listens on TCP & UDP port 53 on all appliance IP addresses. This can be changed if required. For more details, please refer to [Service Socket Addresses](#).

External Health Check Scripts (GSLB)

Custom GSLB health checks can be created and modified using the WebUI.

Adding Health Check Scripts

New scripts can be created either by using the script templates or by uploading files from an external source.

Using Script Templates

To create a new script from template:

1. Using the WebUI, navigate to **Cluster Configuration > Health Check Scripts** and click **Add New Health Check**.

Health Check Details

Name:

GSLB-Custom-Check

?

Type:

GSLB

▼

?

Template:

Example

▼

?

Primary Node Health Check Contents

- Specify an appropriate **Name** for the health check, e.g. **GSLB-Custom-Check**.
- Set **Type** to **GSLB**.
- Using the **Template** dropdown select an appropriate template from the **GSLB** section of the list, e.g. **Example**.
- Modify the script to suit your requirements.
- Click **Update**.

Once the health check has been added, it will appear in the Health Check Scripts list as shown below:


Health Check Scripts

Add New Health Check

Upload Existing Health Check

Health Check Name	Type	In-use	
SMTP	VIP	-	<div>Modify</div> <div>Delete</div>
Ping_IPv4_or_IPv6	VIP	-	<div>Modify</div> <div>Delete</div>
POP3_or_IMAP	VIP	-	<div>Modify</div> <div>Delete</div>
Exchange	VIP	-	<div>Modify</div> <div>Delete</div>
GSLB-Custom-Check	GSLB	-	<div>Modify</div> <div>Delete</div>

The new script will also appear in the **Monitor Script** dropdown when **Monitor** for a Pool is set to **External**:

Monitor	External ▼	?
Monitor Port	80	?
Monitor Script	GSLB-Custom-Check ▼ 	?
Monitor Parameters	GSLB-Custom-Check "arg1","arg2"	?

Uploading External Files

To create a new script by uploading an external file:

1. Using the WebUI, navigate to *Cluster Configuration > Health Check Scripts* and click **Upload Existing Health Check**.

Health Checks - Upload Script

Health check Details

Name:

GSLB-London-Paris-Check

?

Type:

☐ Virtual Service
 ☒ **GSLB**

?

Contents:

Browse...

London-Paris-Check.sh

?

Secondary node contents:

Browse...

No file selected.

?

File is binary:

☐

?

Cancel

Update

2. Specify an appropriate **Name** for the health check, e.g. **GSLB-London-Paris-Check**.
3. Set **Type** to **GSLB**.
4. Click the **Choose File** button next to **Contents**.
5. Browse to and select the required file, e.g. **London-Paris-Check.sh**.

Note

If you have an HA Pair and the secondary node requires a different health check, click the **Choose File** button next to **Secondary Node Contents** and browse to and select the required file.

6. If the file is binary, enable the **File is Binary** checkbox - this will prevent the editor window being displayed.
7. Click **Update**.

Once the health check has been added, it will appear in the Health Check Scripts list and in the **Monitor Script** dropdown as explained in [Using Script Templates](#) above.

GSLB Advanced Configuration

By default (for v8.11 and later), the GSLB service uses the address of the original requester supplied by the EDNS Client Subnet (ECS) option rather than the address of the DNS server that makes the final request to the appliance when considering topologies. If required, this can be disabled so the address of the DNS server is used as with previous versions.

To configure ECS:

1. Using the WebUI, navigate to *GSLB - Advanced Configuration*.



DNS Extensions

Use Client Subnet Extension ☒ ?

Update

2. Configure the *Use Client Subnet Extension* checkbox according to your requirements:
 - If enabled (default), the address of the requesting client will be used when considering Topologies
 - If disabled, the address of the client's DNS server making the request will be used when considering Topologies
3. Click **Update** to apply any changes.

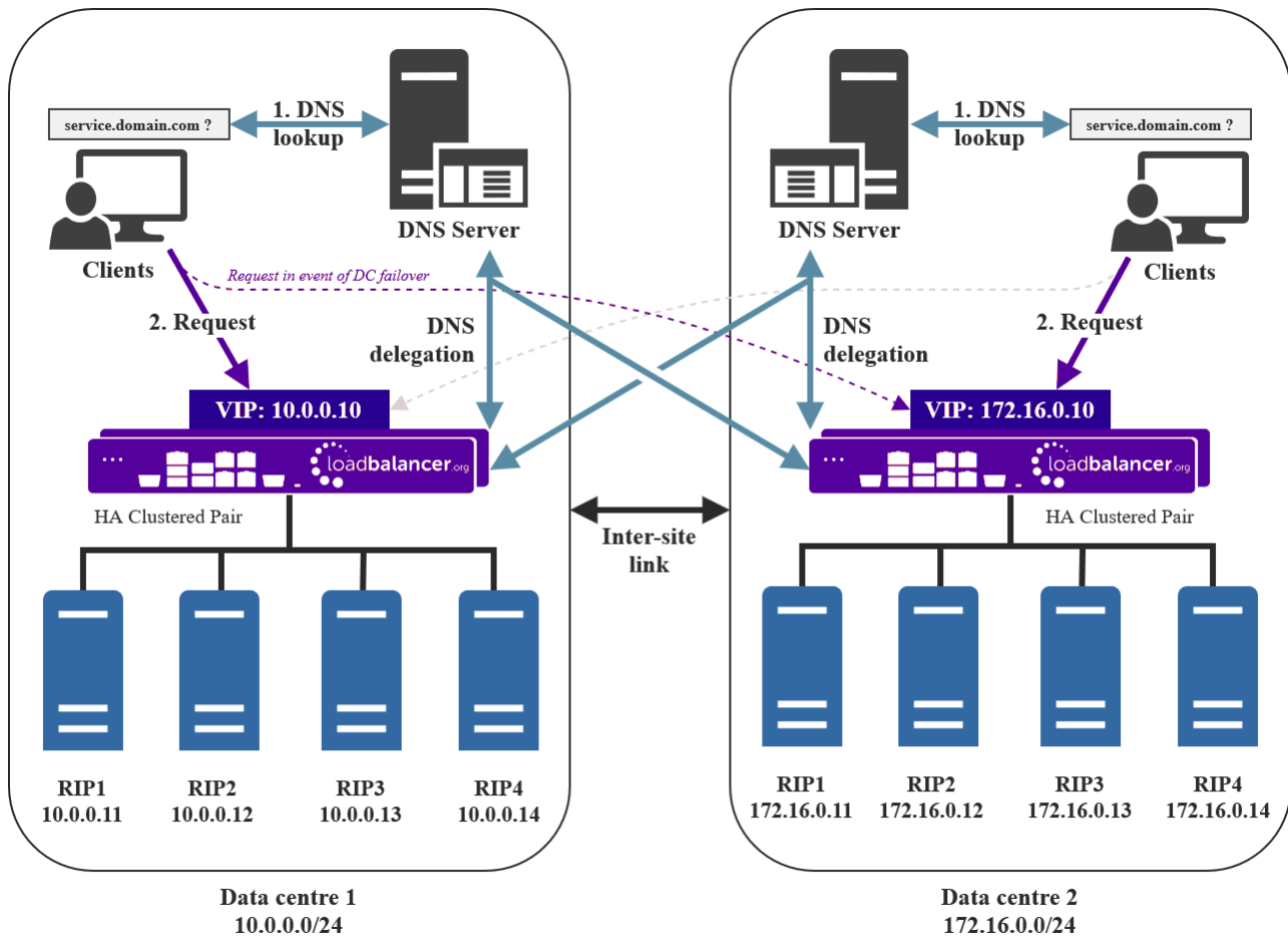
GSLB Multi-site Example

This example demonstrates the steps required to configure GSLB for a two site deployment.

Conceptual Overview

For multi-site deployments, GSLB functionality can be used to provide high availability and location affinity across multiple sites.

- Clients across multiple sites use the same FQDN to access the load balanced service(s)
- Under normal operation, clients are directed to their site's local load balanced cluster (configured using Topology)
- In the event that a site's load balanced service(s) and/or load balancers are offline, then local clients are automatically directed to a functioning load balanced cluster at another site



Explanation:

1. A client tries to access the load balanced service by using the service's FQDN, in this example `service.domain.com`.
2. The client sends a DNS lookup request for **service.domain.com** to its local DNS server.
3. The local site's DNS server has the domain **service.domain.com** delegated to the load balancers.
4. The DNS server sends a delegated DNS lookup request for **service.domain.com** to one of the load balancers.
5. The load balancer that received the delegated DNS lookup request replies to the DNS server by serving up the appropriate, local VIP address. For example, if the request originated from the 10.0.0.0/24 subnet then the VIP in that subnet is served up. Likewise, if the request originated from the 172.16.0.0/24 subnet then the VIP in that subnet is served up. As such, clients are always directed to their local, on-site load balanced service, provided that the on-site instance is online and available and topology has been correctly configured.
6. The DNS server sends the delegated reply to the client.
7. The client connects to the load balanced service at **service.domain.com** by using the local VIP address.

Note

In the event that the load balanced cluster and/or load balancers at one site should completely fail then local clients will be directed to the load balanced cluster at the other site and the service will continue to be available. This style of multi-site failover is possible because the load balancer's GSLB functionality continuously health checks the service at each site. When the service at a site is observed to be unavailable then that site's IP address is no longer served

when responding to DNS queries.

Appliance Configuration

GSLB must be configured on the Primary appliance at each site. The GSLB configuration must be identical at both sites to ensure consistent DNS responses irrespective of which load balancer responds. The following steps assume that an HA pair is already configured in each site. For more information on configuring HA, please refer to [Chapter 9 - Appliance Clustering for HA](#).

Data Center 1

Configure the first HA pair using the WebUI of the Primary appliance in data center 1.

Step 1 – Configure the Global Name

1. Navigate to: *Cluster Configuration > GSLB Configuration*.
2. Select the *Global Names* tab.
3. Click **New Global Name**.

The screenshot shows the 'Global Names' configuration page. The 'Global Names' tab is active. A 'New Global Name' button is located in the top right corner. Below this button is a form titled 'New Global Name'. The form contains three input fields: 'Name' (value: service.domain.com), 'Hostname' (value: service.domain.com), and 'TTL' (value: 0, unit: seconds). Each input field has a help icon (?) to its right. Below the input fields are two buttons: 'Submit' and 'Cancel'. Below the form, the text 'No Data' is displayed.

4. Define the required *Name* and *Hostname*, in this example both are set to **service.domain.com**.
5. Set the *TTL* as required, the default is **0** seconds.
6. Click **Submit**.

Step 2 – Configure the Members

Two members are required, one for each data center.

1. Select the *Members* Tab.
2. Click **New Member**.

Global Names
Members
Pools
Topologies

New Member

New Member

Name	<input type="text" value="nodes-dc1"/>	?
IP	<input type="text" value="10.0.0.10"/>	?
Monitor IP	<input type="text" value="10.0.0.10"/>	?
Weight	<input type="text" value="1"/> ▾	?

Submit
Cancel

No Data

- Enter a **Name** for the first member, e.g. **nodes-dc1**.
- Specify the **IP** and **Monitor IP**, in this example **10.0.0.10**.
- Leave the **Weight** set to **1**.
- Click **submit**.
- To create the second member, click the **New Member** button again and configure the second member (**nodes-dc2,172.16.0.10**) in the same way.

Step 3 – Configure the Pool

- Select the **Pools** Tab.
- Click **New Pool**.

GSLB Configuration

Global Names
Members
Pools
Topologies

New Pool

New Pool

Name	service-nodes	?
Monitor	TCP	?
Monitor Port	80	?
Monitor Send String	check	?
Monitor Match Return	up	?
LB Method	twrr	?
Global Names	service.domain.com	?
Members	<div>Available Members</div> <div>Members In Use</div> <div> nodes-dc1 nodes-dc2 </div>	?

Advanced

Submit
Cancel

No Data

- Enter a suitable **Name** for the Pool, in this example **service-nodes**.
- Set the **Monitor** to **TCP** – this will perform a basic TCP port connect to verify each member.
- Set the **Monitor Port** to the required value, in this example **80**.
- Set the **LB Method** to **twrr** (Topology Weighted Round Robin).
- Select the **Global Name** created previously, e.g. **service.domain.com**.
- Drag the required Members from the **Available Members** list to the **Members in Use** list, in this example **nodes-dc1** and **nodes-dc2**.
- Click **Submit**.



Step 4 – Configure the Topologies

Two Topologies are required, one for each data center.

1. Select the *Topologies* Tab.
2. Click **New Topology**.

GSLB - Configuration

The screenshot shows the 'Topologies' tab in the GSLB configuration interface. A 'New Topology' form is open, containing two input fields: 'Name' with the value 'datacenter1' and 'IP/CIDR' with the value '10.0.0.0/24,10.0.100.0/24,192.168.10.20/32'. There are question mark icons next to each field. At the bottom of the form are 'Submit' and 'Cancel' buttons. A 'New Topology' button is located in the top right corner of the interface. Below the form, the text 'No Data' is displayed.

3. Enter a suitable **Name** for the first Topology, e.g. **datacenter1**.
4. Specify the **IP/CIDR** for the data center, in this example **10.0.0.0/24**.
5. Now add the IP's of clients that will connect to this member under normal circumstances. This can be either as a subnet address or host address, for example **10.0.100.0/24** and **192.168.10.20/32**. Separate each address with a comma as shown.



Note

If you have disabled ECS (see [GSLB Advanced Configuration](#)), the IP address of the DNS servers that are making requests on behalf of clients should be added rather than the client addresses.

6. Click **Submit**.
7. To create the second topology, click the **New Topology** button again and configure the second topology (**datacenter2**) in the same way.

Data Center 2

Now configure the second HA pair in exactly the same way using the WebUI of the Primary appliance in data center 2.



DNS Server Configuration

Once GSLB has been configured at both sites and is identical, the local DNS server at each site must then be configured for GSLB.

The DNS server at each site must be configured to delegate DNS requests for the subdomain in question (in this case **service.domain.com**) to the load balancers. The load balancer's GSLB services will then serve the appropriate A records to the DNS servers and then back to the client making the request.

Using the example presented here, the DNS server at each site would be configured with a delegation for the subdomain **service.domain.com**. The subdomain would be delegated to every load balancer across every site, which provides multi-site redundancy.

The exact steps for creating a DNS delegation vary between different DNS servers and are outside the scope of this document. For further information, a blog post that walks through creating a DNS delegation on a Microsoft DNS server in the context of setting up GSLB on our appliance can be found [here](#) (see the section titled "Delegating your subdomain to your GSLBs using Microsoft's DNS Server").

GSLB Diagnostics

Two reports are available to view the current state of the running GSLB service. These reports are very useful when first setting up GSLB and also when diagnosing any issues. They are available via the WebUI menu option: *Reports*.

GSLB Generic State – This report shows information about the running configuration of GSLB and also the health state of each member/endpoint.

Example:

```
{
  "timestamp": 1658309678.8496737,
  "globalnames": {
    "service.domain.com": {
      "pool_name": "service-nodes",
      "name": "service.domain.com",
      "ttl": 30
    }
  },
  "pools": {
    "service-nodes": {
      "last_status": true,
      "max_addrs_returned": 1,
      "members": [
        {
          "name": "nodes-dc1",
          "status": true,
          "ip": "10.0.0.10",
          "weight": 1,
          "region": "None",
          "status_reason": "monitor passed",
          "retries_left": 2,
          "monitor_ip": "10.0.0.10"
        }
      ]
    }
  }
}
```



```

        "name": "nodes-dc2",
        "status": false,
        "ip": "172.16.0.10",
        "weight": 1,
        "region": "None",
        "status_reason": "OSError [Errno 113] No route to host during
socket.connect()",
        "retries_left": 0,
        "monitor_ip": "172.16.0.10"
    }
],
"name": "service-nodes",
"monitor": {
    "send_string": "None",
    "timeout": 5,
    "name": "tcp",
    "match_re": "None",
    "retries": 2,
    "interval": 10,
    "port": 80
},
"fallback": "any",
"lb_method": "twrr"
}
}
}

```

This shows that:

- Member 10.0.0.10 is passing its health check
- Member 172.16.0.10 is failing its health check
- The running configuration matches the settings configured in the WebUI

GSLB PPDNS State – This report shows information about the running configuration of GSLB and also shows which results will be returned to inbound queries based on the current state of all members/endpoints.

Example:

```

{
  "timestamp": 1658309678.9508624,
  "pools": {
    "service-nodes": {
      "lb_method": "twrr",
      "max_addrs_returned": 1,
      "status": true,
      "fallback": "any",
      "dist_tables": {
        "_default": {
          "index": 0,
          "num_unique_addrs": 1,
          "rotation": [
            "10.0.0.10"
          ]
        }
      }
    }
  }
}

```



```

    }
  },
  "globalnames": {
    "service.domain.com": {
      "pool_name": "service-nodes",
      "ttl": 30
    }
  }
}

```

This shows that:

- Only member 10.0.0.10 is currently in the rotation of addresses being returned because as shown in the **GSLB Generic State** example above, member 172.16.0.10 is failing its health check
- The running configuration matches the settings configured in the WebUI

Note

If you want to configure a multi-site load balanced deployment using GSLB and require further assistance, please don't hesitate to contact support@loadbalancer.org.

Configuring the Appliance via CLI, API & Direct Service Calls

A command line interface (CLI) is included that enables the appliance to be configured and controlled directly from the command line. A JSON based Application Programming Interface (API) is also provided that enables CLI commands to be called via a Web Service. The API is a wrapper around the CLI, so any CLI command can also be called using the API.

It's also possible to directly control layer 4 and layer 7 services, although the disadvantage here is that changes made will not be reflected in the System Overview. If changes are made via the CLI or API, the System Overview is kept in sync.

Command Line Interface (CLI)

All CLI actions are invoked using the **lbcli** command. To obtain help for lbcli, run the following command:

```
lbcli --help
```

A summary of all help options is displayed:

```

lbcli --help sections  list help sections.
lbcli --help syntax    show help for argument syntax.
lbcli --help all       show help for all actions.
lbcli --help [act]     show help for action [act].
lbcli --help about     show information about LBCLI.

```

LBCLI Argument Syntax

All arguments may be prefixed with an ordered, semi-colon separated list of mutations to be applied to the argument within square brackets. Currently the only recognised mutator is **base64** which indicates the value is



base64 encoded and must be decoded before use, for example:

```
lbcli add-certificate --certificate new_cert \  
    --certificate-type pfx \  
    --data '[base64]SGVsbG8gdGhlcmUu'
```

Should a value starting with a [as its first character be desired pass an empty list of mutations first, for example:

```
lbcli add-floating-ip --ip '[][fc00::1234]'
```

All arguments may take their value from a file by giving the @-prefixed filename as the value after any mutation tokens, for example:

```
lbcli edit-healthcheck --label CustomCheck --master @script.sh
```

Should a value with a @ as its first character be desired enter @@ to escape the character. This is only necessary for the first character. To pass a file whose name starts with a @ character it is necessary to include the path to break up the @@ sequence for example:

```
lbcli edit-csr --csr MainCert --crt-body @./@cert.pem
```

Both mutators and file input may be used at the same time, for example:

```
lbcli set-snmp --community '[base64]@snmpname.b64'
```

LBCLI Command Reference

Activate CA Certificate

```
lbcli activate-ca-certificate
```

Make a CA certificate active within its Family

Required Arguments

```
--family  expects an argument of type existing-cacert-family  
--cert    expects an argument of type existing-cacert
```

Add an ACL rule

```
lbcli add-acl  
or  
lbcli --action acl --function add
```

Adds an ACL rule to the specified layer 7 virtual service. If allow-duplicates is on then the new ACL will always be



appended even if there is an existing, matching ACL defined already.

Required Arguments

<code>--vip</code>	expects an argument of type <code>existing-l7-vip</code>
<code>--allow-duplicates</code>	expects an argument of type <code>bool</code> defaults to <code>off</code>
<code>--bool</code>	argument should be one of <code>equal</code> or <code>notEqual</code> defaults to <code>equal</code>
<code>--denystatus</code>	argument should be one of <code>200</code> , <code>400</code> , <code>403</code> , <code>405</code> , <code>408</code> , <code>425</code> , <code>429</code> , <code>500</code> , <code>502</code> , <code>503</code> or <code>504</code> defaults to <code>403</code>
<code>--freetype</code>	expects an argument of type <code>string</code> defaults to
<code>--headername</code>	expects an argument of type <code>string</code> defaults to
<code>--location</code>	expects an argument of type <code>string</code> defaults to
<code>--path</code>	expects an argument of type <code>string</code> defaults to <code>/</code>
<code>--pathtype</code>	argument should be one of <code>path_beg</code> , <code>path_end</code> , <code>path_reg</code> , <code>hdr_host</code> , <code>hdr_beg</code> , <code>query</code> , <code>src_blk</code> , <code>header</code> , <code>path</code> , <code>sni</code> , <code>ssl_sni</code> , <code>dst_port</code> , <code>flag</code> or <code>freetype</code>
<code>--redirecttype</code>	argument should be one of <code>url_loc</code> , <code>url_pre</code> , <code>backend</code> , <code>drop</code> , <code>none</code> , <code>use_server</code> or <code>flag</code>

Add CA Certificate

```
lbcli add-ca-certificate
```

Add a CA Certificate to a Family

Required Arguments

<code>--family</code>	expects an argument of type <code>existing-cacert-family</code>
<code>--contents</code>	expects an argument of type <code>string</code>
<code>--cert</code>	expects an argument of type <code>strict-label</code>

Optional Arguments

```
--auto-fix-label expects an argument of type bool
```

Add CA Certificate Family

```
lbcli add-ca-certificate-family
```

Add a CA Certificate Family

Required Arguments

<code>--family</code>	expects an argument of type <code>strict-label</code>
<code>--cert</code>	expects an argument of type <code>strict-label</code>
<code>--contents</code>	expects an argument of type <code>string</code>

Optional Arguments

```
--crl-contents expects an argument of type string
--auto-fix-label expects an argument of type bool
```



Add an SSL certificate

```
lbcli add-certificate
```

Add an SSL certificate to this cluster

Required Arguments

<code>--certificate</code>	expects an argument of type strict-label
<code>--data</code>	expects an argument of type string

Optional Arguments

<code>--certificate-type</code>	argument should be one of pem or pfx defaults to pem
<code>--password</code>	expects an argument of type string defaults to

Create a Certificate Signing Request

```
lbcli add-csr
or
lbcli --action termination --function csr --type certificate
```

Create a Certificate Signing Request. When `--yes-i-am-sure` on is specified this command will overwrite existing certificates.

Required Arguments

<code>--country</code>	expects an argument of type country-code
<code>--province</code>	expects an argument of type string
<code>--city</code>	expects an argument of type string
<code>--organisation</code>	expects an argument of type string
<code>--unit</code>	expects an argument of type string
<code>--domain</code>	expects an argument of type string
<code>--email</code>	expects an argument of type string
<code>--csr-size,</code> <code>--csrsize</code>	argument should be one of 2048 or 4096 defaults to 4096
<code>--sign-algorithm,</code> <code>--signalgorithm</code>	argument should be one of sha256 defaults to sha256
<code>--certificate,</code> <code>--csr-name,</code> <code>--csrname</code>	expects an argument of type string

Optional Arguments

<code>--yes-i-am-sure</code>	expects an argument of type bool
<code>--subject-alternative-name</code>	expects an argument of type string

Add a floating IP address

```
lbcli add-floating-ip
```



Adds a floating IP to the cluster.

Required Arguments

```
--ip          expects an argument of type ip
```

Optional Arguments

```
--disabled    expects an argument of type bool defaults to off
```

Add a global name

```
lbcli add-gslb-globalname
or
lbcli --action gslb --function add --section globalnames
```

Adds a GSLB global name.

Required Arguments

```
--name        expects an argument of type string
--hostname     expects an argument of type hostname
--ttl         expects an argument of type uint
```

Add a member

```
lbcli add-gslb-member
or
lbcli --action gslb --function add --section members
```

Adds a GSLB member.

Required Arguments

```
--name        expects an argument of type string
--ip          expects an argument of type ip
--weight      expects an argument of type gslb-weight
```

Optional Arguments

```
--monitor-ip  argument should be one of ip-or-empty
--add-pool    expects an argument of type string
```

Add a pool

```
lbcli add-gslb-pool
or
lbcli --action gslb --function add --section pools
```

Adds a GSLB pool.



Required Arguments

<code>--add-globalname</code>	expects an argument of type string
<code>--lb-method</code>	argument should be one of wrr, twrr or fogroup
<code>--monitor</code>	argument should be one of http, tcp, forced, external, external_weight or externaldynamicweight
<code>--name</code>	expects an argument of type string

Optional Arguments

<code>--add-member</code>	expects an argument of type string
<code>--fallback</code>	argument should be one of any or refuse defaults to any
<code>--max-addrs-returned</code>	expects an argument of type uint defaults to 1
<code>--monitor-expected-codes</code>	expects an argument of type http-code-list
<code>--monitor-hostname</code>	argument should be one of hostname-or-empty
<code>--monitor-interval</code>	expects an argument of type uint defaults to 10
<code>--monitor-parameters</code>	expects an argument of type string
<code>--monitor-port</code>	expects an argument of type port
<code>--monitor-result</code>	expects an argument of type string
<code>--monitor-retries</code>	expects an argument of type uint defaults to 2
<code>--monitor-return-match</code>	expects an argument of type string
<code>--monitor-script</code>	expects an argument of type string
<code>--monitor-send-string</code>	expects an argument of type string
<code>--monitor-status</code>	argument should be one of up or down
<code>--monitor-timeout</code>	expects an argument of type uint defaults to 5000
<code>--monitor-url-path</code>	expects an argument of type string
<code>--monitor-use-ssl</code>	expects an argument of type bool

Add a topology

```
lbcli add-gslb-topology
or
lbcli --action gslb --function add --section topologies
```

Adds a GSLB topology.

Required Arguments

<code>--name</code>	expects an argument of type string
---------------------	------------------------------------

Optional Arguments

<code>--add-ips</code>	expects an argument of type string
------------------------	------------------------------------

Add HAProxy SSL Associated Termination

```
lbcli add-haproxy-associated-termination
```

Add an HAProxy SSL Associated Termination to a Virtual Service



Required Arguments

<code>--vip</code>	expects an argument of type existing-vip
<code>--certificate</code>	expects an argument of type strict-label
<code>--ca-certificate</code>	argument should be one of strict-label-or-empty defaults to
<code>--port</code>	expects an argument of type port defaults to 443
<code>--disablesslv3</code>	expects an argument of type bool defaults to on
<code>--disabletlsv1</code>	expects an argument of type bool defaults to on
<code>--disabletlsv1-1</code>	expects an argument of type bool defaults to on
<code>--disabletlsv1-2</code>	expects an argument of type bool defaults to off
<code>--disabletlsv1-3</code>	expects an argument of type bool defaults to off

Optional Arguments

<code>--ciphers</code>	expects an argument of type cipher-list
<code>--alpn</code>	expects an argument of type alpn-list

Add a header

```
lbcli add-header
or
lbcli --action headers --function add
```

Adds a header rule to the specified layer 7 virtual service.

Required Arguments

<code>--header-type</code>	argument should be one of http-request or http-response
<code>--vip</code>	expects an argument of type existing-l7-vip
<code>--header-option</code>	argument should be one of add, set, del, delete or replace
<code>--header-name</code>	expects an argument of type string
<code>--header-value</code>	expects an argument of type string

Add a health check

```
lbcli add-healthcheck
```

Add a health check to the cluster. If the `--slave` argument is omitted then the script passed to the `--master` argument will be used on both appliances. If `--base64` is yes then the health check body should be base64 encoded. If `--zlib` is set then the health check body should be compressed according to RFC-1950. If both base64 encoding and compression are used then the body should first be compressed then base64 encoded.

Required Arguments

<code>--label</code>	expects an argument of type string
<code>--kind</code>	argument should be one of check or gslbcheck defaults to check
<code>--master</code>	expects an argument of type string

Optional Arguments



```
--base64  expects an argument of type bool defaults to no
--zlib    expects an argument of type bool defaults to no
--slave   expects an argument of type string
```

Add a layer 4 real server

```
lbcli add-layer4-server
or
lbcli --action add-rip --layer 4
```

Adds a layer 4 real server to the specified virtual service.

Required Arguments

```
--vip      expects an argument of type existing-vip
--ip       expects an argument of type ip
--rip      expects an argument of type rip
```

Optional Arguments

```
--port      expects an argument of type port
--weight     expects an argument of type rip-weight defaults to 100
--minconns  expects an argument of type uint
--maxconns  expects an argument of type uint
```

Add a layer 4 virtual service

```
lbcli add-layer4-service
or
lbcli --action add-vip --layer 4
```

Adds a layer 4 virtual service to the cluster.

Required Arguments

```
--ip       expects an argument of type ip
--ports    argument should be one of port-list-or-asterisk
--vip      expects an argument of type vip
```

Optional Arguments

```
--forwarding  argument should be one of gate, masq, ipip or snat
               defaults to gate
--protocol    argument should be one of tcp, udp, tcpudp, ops or fwm
               defaults to tcp
--slave-ip    expects an argument of type ip
--granularity expects an argument of type cidr
--fallback-ip expects an argument of type ip
--fallback-port expects an argument of type port
--persistent  expects an argument of type bool
--persist-time expects an argument of type uint
--scheduler   argument should be one of wlc, wrr or dh
```



<code>--feedback</code>	argument should be one of agent, http or none
<code>--email</code>	expects an argument of type string
<code>--email-from</code>	expects an argument of type string
<code>--check-service</code>	argument should be one of none, dns, ftp, http, http_proxy, https, imap, imaps, ldap, mysql, nntp, pop, pops, radius, simpletcp, sip or smtp
<code>--check-vhost</code>	expects an argument of type string
<code>--check-database</code>	expects an argument of type string
<code>--check-login</code>	expects an argument of type string
<code>--check-password</code>	expects an argument of type string
<code>--check-type</code>	argument should be one of negotiate, connect, ping, external, off, on, 5 or 10
<code>--check-port</code>	expects an argument of type port
<code>--check-request</code>	expects an argument of type string
<code>--check-response</code>	expects an argument of type string
<code>--check-secret</code>	expects an argument of type string
<code>--check-command</code>	expects an argument of type string
<code>--feedback-port</code>	expects an argument of type port
<code>--fallback-local</code>	expects an argument of type bool
<code>--autoscale-group</code>	expects an argument of type string

Add a layer 7 real server

```
lbcli add-layer7-server
or
lbcli --action add-rip --layer 7
```

Adds a layer 7 real server to the specified virtual service.

Required Arguments

<code>--vip</code>	expects an argument of type existing-vip
<code>--ip</code>	expects an argument of type ip
<code>--rip</code>	expects an argument of type rip

Optional Arguments

<code>--port</code>	expects an argument of type port
<code>--weight</code>	expects an argument of type rip-weight defaults to 100
<code>--minconns</code>	expects an argument of type uint
<code>--maxconns</code>	expects an argument of type uint
<code>--encrypted</code>	expects an argument of type bool defaults to off
<code>--cert</code>	argument should be one of existing-certificate-or-empty defaults to
<code>--cacert</code>	argument should be one of existing-cacert-family-or-empty defaults to
<code>--redir</code>	expects an argument of type string
<code>--redir-enabled</code>	expects an argument of type bool defaults to off

Add a layer 7 virtual service

```
lbcli add-layer7-service
or
lbcli --action add-vip --layer 7
```



Adds a layer 7 virtual service to the cluster.

Required Arguments

<code>--disablesslv3</code>	expects an argument of type bool defaults to true
<code>--disabletlsv1</code>	expects an argument of type bool defaults to true
<code>--disabletlsv1-1</code>	expects an argument of type bool defaults to false
<code>--disabletlsv1-2</code>	expects an argument of type bool defaults to false
<code>--disabletlsv1-3</code>	expects an argument of type bool defaults to false
<code>--ip</code>	expects an argument of type ip
<code>--ports</code>	expects an argument of type port-list
<code>--vip</code>	expects an argument of type vip

Optional Arguments

<code>--ca-certificate</code>	argument should be one of existing-cacert-family-or-empty
<code>--certificate</code>	expects an argument of type existing-certificate
<code>--ciphers</code>	expects an argument of type cipher-list
<code>--mode</code>	argument should be one of other_tcp, tcp, ip, http or http2 defaults to http
<code>--fallback-ip</code>	expects an argument of type ip
<code>--fallback-port</code>	expects an argument of type port
<code>--service-type</code>	expects an argument of type string
<code>--slave-ip</code>	expects an argument of type ip
<code>--no-write</code>	expects an argument of type bool
<code>--backend-only</code>	expects an argument of type bool
<code>--appsession-cookie</code>	expects an argument of type string
<code>--as-port</code>	expects an argument of type port
<code>--autoscale-group</code>	expects an argument of type string
<code>--backend-encryption</code>	expects an argument of type bool
<code>--check-host</code>	expects an argument of type string
<code>--check-negate-response</code>	expects an argument of type bool
<code>--check-password</code>	expects an argument of type string
<code>--check-port</code>	argument should be one of port-or-empty
<code>--check-receive</code>	expects an argument of type string
<code>--check-request</code>	expects an argument of type string
<code>--check-type</code>	argument should be one of connect, external, negotiate_http, negotiate_https, negotiate_http_head, negotiate_https_head, negotiate_http_options, negotiate_https_options, mysql or none
<code>--check-username</code>	expects an argument of type string
<code>--clear-stick-drain</code>	expects an argument of type bool
<code>--compression</code>	expects an argument of type bool
<code>--cookie-attr-domain</code>	expects an argument of type space-seperated-cookie-domains
<code>--cookie-attr-httponly</code>	expects an argument of type bool
<code>--cookie-attr-secure</code>	expects an argument of type bool
<code>--cookie-maxidle</code>	expects an argument of type haproxy-interval
<code>--cookie-maxlife</code>	expects an argument of type haproxy-interval

<code>--cookie-name</code>	expects an argument of type string
<code>--default-ciphers</code>	argument should be one of cipher-list-or-empty
<code>--enable-hsts</code>	expects an argument of type bool
<code>--encrypt-all-backends</code>	expects an argument of type bool
<code>--external-check-script</code>	expects an argument of type existing-healthcheck
<code>--fallback-disabled</code>	expects an argument of type bool
<code>--fallback-encrypt</code>	expects an argument of type bool
<code>--fallback-persist</code>	expects an argument of type bool
<code>--feedback-method</code>	argument should be one of agent or none
<code>--feedback-port</code>	expects an argument of type port
<code>--force-to-https</code>	expects an argument of type bool
<code>--force-to-https-port</code>	argument should be one of port-or-empty
<code>--forward-for</code>	expects an argument of type bool
<code>--hsts-month</code>	expects an argument of type uint
<code>--http-pipeline</code>	argument should be one of http_keep_alive, http_close, http_server_close or http_force_close
<code>--http-pretend-keepalive</code>	expects an argument of type bool
<code>--http-request</code>	expects an argument of type bool
<code>--invalid-http</code>	expects an argument of type bool
<code>--maxconn</code>	expects an argument of type uint
<code>--persist-table-size</code>	expects an argument of type uint
<code>--persist-time</code>	expects an argument of type uint
<code>--persistence</code>	argument should be one of none, ip, tcp, sslsesid, http, appsession, rdp-session, rdp-cookie, http_ip, waf, last, xff or fallback_persist
<code>--proxy-bind</code>	expects an argument of type string
<code>--redirect-code</code>	expects an argument of type http-300-code
<code>--redispatch</code>	expects an argument of type bool
<code>--reuse-idle</code>	expects an argument of type bool
<code>--scheduler</code>	argument should be one of first, roundrobin or leastconn
<code>--send-proxy</code>	argument should be one of none, v1, v2, v2_ssl or v2_ssl_cn defaults to none
<code>--send-rip-name-as-host</code>	expects an argument of type bool
<code>--source-address</code>	expects an argument of type ip
<code>--stunnel-source</code>	expects an argument of type string
<code>--stunnelproxy</code>	expects an argument of type bool
<code>--tcp-keep-alive</code>	expects an argument of type bool
<code>--timeout</code>	expects an argument of type bool
<code>--timeout-client</code>	expects an argument of type haproxy-interval
<code>--timeout-server</code>	expects an argument of type haproxy-interval
<code>--tproxy</code>	expects an argument of type bool
<code>--tunneltimeout</code>	expects an argument of type haproxy-interval
<code>--use-content-inspection</code>	expects an argument of type bool
<code>--verify-client-certificate</code>	argument should be one of required, optional or none defaults to required
<code>--xff-ip-pos</code>	expects an argument of type int

Add a PBR rule

```
lbcli add-pbr
or
lbcli --action pbr --function set
```



```
lbcli set-pbr
```

Adds a policy based routing rule to the appliance.

Required Arguments

```
--ip           expects an argument of type ip
--gateway      expects an argument of type ip
```

Optional Arguments

```
--local-routing expects an argument of type bool defaults to off
```

Add a Certificate Revocation List

```
lbcli add-revocation-list
```

Add or replace the Certificate Revocation List for a Family

Required Arguments

```
--family      expects an argument of type existing-cacert-family
--contents    expects an argument of type string
```

Add a static IP address

```
lbcli add-static-ip
or
lbcli --action address --function add
```

Adds a static IP to the specified virtual service.

Required Arguments

```
--interface  expects an argument of type existing-interface
--address    expects an argument of type ip-with-optional-cidr
```

Optional Arguments

```
--cidr       expects an argument of type cidr
```

Add a static route

```
lbcli add-static-route
or
lbcli --action route --function static --type add
```

Adds a route via the given gateway to the specified network.



Required Arguments

```
--gateway expects an argument of type ip
--network expects an argument of type ip-with-optional-cidr
```

Optional Arguments

```
--cidr expects an argument of type cidr
```

Add an SSL Termination

```
lbcli add-termination
or
lbcli --action termination --function add --type stunnel
```

Adds an SSL Termination to the cluster.

Required Arguments

```
--port expects an argument of type port
--disablesslv2 expects an argument of type bool defaults to true
--disablesslv3 expects an argument of type bool defaults to true
--disabletlsv1 expects an argument of type bool defaults to true
--stunnelnsdelay expects an argument of type bool defaults to true
--stunnelproxy expects an argument of type bool defaults to false
--servercipherorder expects an argument of type bool defaults to true
--emptyfragments expects an argument of type bool defaults to true
--stunnelrenegotiation expects an argument of type bool defaults to true
--stunneltimetoclose expects an argument of type uint defaults to 0
--sslcert expects an argument of type existing-certificate
defaults to server
--proxy-bind expects an argument of type bool defaults to false
--disabletlsv1-1 expects an argument of type bool defaults to false
--disabletlsv1-2 expects an argument of type bool defaults to false
--disabletlsv1-3 expects an argument of type bool defaults to false
--sslmode argument should be one of high, fips or compatable
defaults to high
```

Optional Arguments

```
--vip expects an argument of type string
--associated-to expects an argument of type existing-l7-vip
--proxy-bind expects an argument of type existing-l7-vip
--source expects an argument of type ip
--slave-source expects an argument of type ip
--ciphers expects an argument of type cipher-list
--ip expects an argument of type ip
--slave-ip expects an argument of type ip
--backend-ip expects an argument of type ip
--slave-backend-ip expects an argument of type ip
--backend-port expects an argument of type port
--stunnel-source expects an argument of type string
```

Add a WAF



```
lbcli add-waf
```

Adds a Web Application Firewall to the specified layer 7 virtual service.

Required Arguments

```
--waf    expects an argument of type string
--vip    expects an argument of type existing-l7-vip
```

Clone a health check

```
lbcli clone-healthcheck
```

Clone an existing health check on the cluster.

Required Arguments

```
--label  expects an argument of type existing-healthcheck
```

Clone a virtual service.

```
lbcli clone-layer4-service
or
lbcli --action clone-vip --layer 4
```

Create a copy of the specified virtual service.

Required Arguments

```
--vip      expects an argument of type existing-vip
--ip       expects an argument of type ip
--ports    argument should be one of port-list-or-asterisk
--clone    expects an argument of type label
```

Optional Arguments

```
--forwarding  argument should be one of gate, masq, ipip or snat
--protocol    argument should be one of tcp, udp, tcpudp, ops or fwm
--slave-ip    expects an argument of type ip
--granularity expects an argument of type cidr
--fallback-ip expects an argument of type ip
--fallback-port expects an argument of type port
--persistent  expects an argument of type bool
--persist-time expects an argument of type uint
--scheduler   argument should be one of wlc, wrp or dh
--feedback    argument should be one of agent, http or none
--email       expects an argument of type string
--email-from  expects an argument of type string
--check-service argument should be one of none, dns, ftp, http,
             http_proxy, https, imap, imaps, ldap, mysql, nntp, pop,
             pops, radius, simpletcp, sip or smtp
--check-vhost expects an argument of type string
```



<code>--check-database</code>	expects an argument of type string
<code>--check-login</code>	expects an argument of type string
<code>--check-password</code>	expects an argument of type string
<code>--check-type</code>	argument should be one of negotiate, connect, ping, external, off, on, 5 or 10
<code>--check-port</code>	expects an argument of type port
<code>--check-request</code>	expects an argument of type string
<code>--check-response</code>	expects an argument of type string
<code>--check-secret</code>	expects an argument of type string
<code>--check-command</code>	expects an argument of type string
<code>--feedback-port</code>	expects an argument of type port
<code>--fallback-local</code>	expects an argument of type bool
<code>--autoscale-group</code>	expects an argument of type string

Clone a virtual service.

```
lbcli clone-layer7-service
or
lbcli --action clone-vip --layer 7
```

Create a copy of the specified virtual service.

Required Arguments

<code>--vip</code>	expects an argument of type existing-vip
<code>--ip</code>	expects an argument of type ip
<code>--ports</code>	expects an argument of type port-list
<code>--clone</code>	expects an argument of type label

Optional Arguments

<code>--appsession-cookie</code>	expects an argument of type string
<code>--as-port</code>	expects an argument of type port
<code>--autoscale-group</code>	expects an argument of type string
<code>--backend-encryption</code>	expects an argument of type bool
<code>--backend-only</code>	expects an argument of type bool
<code>--ca-certificate</code>	argument should be one of existing-cacert-family-or-empty
<code>--certificate</code>	expects an argument of type existing-certificate
<code>--check-host</code>	expects an argument of type string
<code>--check-negate-response</code>	expects an argument of type bool
<code>--check-password</code>	expects an argument of type string
<code>--check-port</code>	argument should be one of port-or-empty
<code>--check-receive</code>	expects an argument of type string
<code>--check-request</code>	expects an argument of type string
<code>--check-type</code>	argument should be one of connect, external, negotiate_http, negotiate_https, negotiate_http_head, negotiate_https_head, negotiate_http_options, negotiate_https_options, mysql or none
<code>--check-username</code>	expects an argument of type string
<code>--ciphers</code>	expects an argument of type cipher-list
<code>--clear-stick-drain</code>	expects an argument of type bool
<code>--compression</code>	expects an argument of type bool
<code>--cookie-attr-domain</code>	expects an argument of type



<code>--cookie-attr-httponly</code>	space-seperated-cookie-domains
<code>--cookie-attr-secure</code>	expects an argument of type bool
<code>--cookie-maxidle</code>	expects an argument of type bool
<code>--cookie-maxlife</code>	expects an argument of type haproxy-interval
<code>--cookiename</code>	expects an argument of type haproxy-interval
<code>--default-ciphers</code>	expects an argument of type string
	argument should be one of
	cipher-list-or-empty
<code>--disablesslv3</code>	expects an argument of type bool
<code>--disabletls1</code>	expects an argument of type bool
<code>--disabletls1-1</code>	expects an argument of type bool
<code>--disabletls1-2</code>	expects an argument of type bool
<code>--disabletls1-3</code>	expects an argument of type bool
<code>--enable-hsts</code>	expects an argument of type bool
<code>--encrypt-all-backends</code>	expects an argument of type bool
<code>--external-check-script</code>	expects an argument of type
	existing-healthcheck
<code>--fallback-disabled</code>	expects an argument of type bool
<code>--fallback-encrypt</code>	expects an argument of type bool
<code>--fallback-ip</code>	expects an argument of type ip
<code>--fallback-persist</code>	expects an argument of type bool
<code>--fallback-port</code>	expects an argument of type port
<code>--feedback-method</code>	argument should be one of agent or none
<code>--feedback-port</code>	expects an argument of type port
<code>--force-to-https</code>	expects an argument of type bool
<code>--force-to-https-port</code>	argument should be one of port-or-empty
<code>--forward-for</code>	expects an argument of type bool
<code>--hsts-month</code>	expects an argument of type uint
<code>--http-pipeline</code>	argument should be one of http_keep_alive,
	http_close, http_server_close or
	http_force_close
<code>--http-pretend-keepalive</code>	expects an argument of type bool
<code>--http-request</code>	expects an argument of type bool
<code>--invalid-http</code>	expects an argument of type bool
<code>--maxconn</code>	expects an argument of type uint
<code>--mode</code>	argument should be one of other_tcp, tcp, ip,
	http or http2
<code>--no-write</code>	expects an argument of type bool
<code>--persist-table-size</code>	expects an argument of type uint
<code>--persist-time</code>	expects an argument of type uint
<code>--persistence</code>	argument should be one of none, ip, tcp,
	sslsesid, http, appsession, rdp-session,
	rdp-cookie, http_ip, waf, last, xff or
	fallback_persist
<code>--proxy-bind</code>	expects an argument of type string
<code>--redirect-code</code>	expects an argument of type http-300-code
<code>--redispatch</code>	expects an argument of type bool
<code>--reuse-idle</code>	expects an argument of type bool
<code>--scheduler</code>	argument should be one of roundrobin,
	leastconn or first
<code>--send-proxy</code>	argument should be one of none, v1, v2,
	v2_ssl or v2_ssl_cn defaults to none
<code>--send-rip-name-as-host</code>	expects an argument of type bool
<code>--slave-ip</code>	expects an argument of type ip
<code>--source-address</code>	expects an argument of type ip
<code>--stunnel-source</code>	expects an argument of type string
<code>--stunneltpoxy</code>	expects an argument of type bool
<code>--tcp-keep-alive</code>	expects an argument of type bool
<code>--timeout</code>	expects an argument of type bool

<code>--timeout-client</code>	expects an argument of type haproxy-interval
<code>--timeout-server</code>	expects an argument of type haproxy-interval
<code>--tproxy</code>	expects an argument of type bool
<code>--tunneltimeout</code>	expects an argument of type haproxy-interval
<code>--use-content-inspection</code>	expects an argument of type bool
<code>--verify-client-certificate</code>	argument should be one of required, optional or none
<code>--xff-ip-pos</code>	expects an argument of type int

Delete an ACL rule

```
lbcli delete-acl
or
lbcli --action acl --function del
lbcli --action acl --function delete
lbcli del-acl
```

Delete the first matching ACL rule from the specified layer 7 virtual service.

Required Arguments

<code>--vip</code>	expects an argument of type existing-l7-vip
<code>--bool</code>	argument should be one of equal or notEqual defaults to equal
<code>--denystatus</code>	argument should be one of 200, 400, 403, 405, 408, 425, 429, 500, 502, 503 or 504 defaults to 403
<code>--freetype</code>	expects an argument of type string defaults to
<code>--headername</code>	expects an argument of type string defaults to
<code>--location</code>	expects an argument of type string defaults to
<code>--path</code>	expects an argument of type string defaults to /
<code>--pathtype</code>	argument should be one of path_reg, path_beg, path_end, hdr_host, hdr_beg, query, src_blk, header, path, sni, ssl_sni, dst_port, flag or freetype
<code>--redirecttype</code>	argument should be one of url_loc, url_pre, backend, drop, none or use_server

Delete CA Certificate

```
lbcli delete-ca-certificate
or
lbcli del-ca-certificate
```

Remove a CA certificate from its Family

Required Arguments

<code>--family</code>	expects an argument of type existing-cacert-family
<code>--cert</code>	expects an argument of type existing-cacert

Delete CA Certificate Family

```
lbcli delete-ca-certificate-family
or
```



```
lbcli del-ca-certificate-family
```

Delete a CA Certificate Family

Required Arguments

```
--family expects an argument of type existing-cacert-family
```

Delete an SSL certificate

```
lbcli delete-certificate
```

Delete an SSL certificate to this cluster

Required Arguments

```
--certificate expects an argument of type existing-certificate
```

Delete a floating IP

```
lbcli delete-floating-ip  
or  
lbcli del-floating-ip
```

Delete a floating IP from the cluster.

Required Arguments

```
--ip expects an argument of type ip
```

Delete a global name

```
lbcli delete-gslb-globalname  
or  
lbcli --action gslb --function del --section globalnames  
lbcli --action gslb --function delete --section globalnames  
lbcli del-gslb-globalname
```

Delete the specified GSLB global name.

Required Arguments

```
--name expects an argument of type string
```

Delete a member

```
lbcli delete-gslb-member  
or  
lbcli --action gslb --function del --section members  
lbcli --action gslb --function delete --section members
```



```
lbcli del-gslb-member
```

Delete the specified GSLB member.

Required Arguments

```
--name expects an argument of type string
```

Delete a pool

```
lbcli delete-gslb-pool
or
lbcli --action gslb --function del --section pools
lbcli --action gslb --function delete --section pools
lbcli del-gslb-pool
```

Delete the specified GSLB pool.

Required Arguments

```
--name expects an argument of type string
```

Delete a member

```
lbcli delete-gslb-topology
or
lbcli --action gslb --function del --section topologies
lbcli --action gslb --function delete --section topologies
lbcli del-gslb-topology
```

Delete the specified GSLB member.

Required Arguments

```
--name expects an argument of type string
```

Delete HAProxy SSL Associated Termination

```
lbcli delete-haproxy-associated-termination
or
lbcli del-haproxy-associated-termination
```

Delete a HAProxy SSL Associated Termination from Virtual Service

Required Arguments

```
--vip expects an argument of type existing-vip
```

Delete a header rule



```
lbcli delete-header
or
lbcli --action headers --function delete
lbcli --action headers --function del
lbcli del-header
```

Delete the first matching header rule from the specified layer 7 virtual service.

Required Arguments

```
--vip          expects an argument of type existing-l7-vip
--header-option argument should be one of add, set, del, delete or
               replace
--header-name   expects an argument of type string
```

Delete a health check

```
lbcli delete-healthcheck
or
lbcli del-healthcheck
```

Delete an existing health check from the cluster.

Required Arguments

```
--label expects an argument of type existing-healthcheck
```

Delete a header rule

```
lbcli delete-pbr
or
lbcli --action pbr --function del
lbcli --action pbr --function delete
lbcli del-pbr
```

Delete the first matching header rule from the specified layer 7 virtual service.

Required Arguments

```
--ip          expects an argument of type ip
--gateway     expects an argument of type ip
```

Delete a Certificate Revocation List

```
lbcli delete-revocation-list
or
lbcli del-revocation-list
```

Delete Certificate Revocation List from a Family



Required Arguments

```
--family expects an argument of type existing-cacert-family
```

Delete a real server

```
lbcli delete-server  
or  
lbcli del-rip  
lbcli delete-rip
```

Delete the real server from the specified virtual service.

Required Arguments

```
--vip expects an argument of type existing-vip  
--rip expects an argument of type existing-rip
```

Delete a virtual service

```
lbcli delete-service  
or  
lbcli del-service  
lbcli del-vip  
lbcli delete-vip
```

Delete a virtual service from the cluster.

Required Arguments

```
--vip expects an argument of type existing-vip
```

Delete a static IP address

```
lbcli delete-static-ip  
or  
lbcli --action address --function delete  
lbcli --action address --function del  
lbcli del-static-ip
```

Delete a static IP from the appliance.

Required Arguments

```
--interface expects an argument of type existing-interface  
--address expects an argument of type ip-with-optional-cidr
```

Optional Arguments

```
--cidr expects an argument of type cidr
```



Delete a static route

```
lbcli delete-static-route
or
lbcli --action route --function static --type del
lbcli --action route --function static --type delete
lbcli del-static-route
```

Delete a static route from the appliance.

Required Arguments

```
--network expects an argument of type ip-with-optional-cidr
```

Optional Arguments

```
--cidr expects an argument of type cidr
```

Delete an SSL termination

```
lbcli delete-termination
or
lbcli --action termination --function del --type stunnel
lbcli --action termination --function delete --type stunnel
lbcli del-termination
```

Delete an SSL termination from the appliance.

Required Arguments

```
--vip expects an argument of type existing-termination
```

Delete a WAF

```
lbcli delete-waf
```

Delete the specified Web Application Firewall from the cluster.

Required Arguments

```
--waf expects an argument of type existing-waf
--vip expects an argument of type existing-vip
```

Disable the API endpoint

```
lbcli disable-api
or
lbcli --action api --function disable
```

Disable the API endpoint on the appliance.



Disable Floating IP.

```
lbcli disable-floating-ip
```

Make it so that the given floating IP cannot be made active on this appliance.

Drain a real server

```
lbcli drain
```

Drain connections from the specified real server. No new connections will be allowed to the real server but existing connections will remain until they are closed.

Required Arguments

```
--vip    expects an argument of type existing-vip
--rip    expects an argument of type existing-rip
```

Edit the sections of a CSR

```
lbcli edit-csr
```

Edit the sections of a CSR

Required Arguments

```
--certificate  expects an argument of type existing-certificate
```

Optional Arguments

```
--crt-body    expects an argument of type string
--int-body    expects an argument of type string
--root-body    expects an argument of type string
```

Edit a global name

```
lbcli edit-gslb-globalname
or
lbcli --action gslb --function edit --section globalnames
```

Edit a GSLB global name.

Required Arguments

```
--name        expects an argument of type string
--hostname    expects an argument of type hostname
--ttl         expects an argument of type uint
```

Optional Arguments



```
--new-name expects an argument of type string
```

Edit a member

```
lbcli edit-gslb-member
or
lbcli --action gslb --function edit --section members
```

Edit a GSLB member.

Required Arguments

```
--name expects an argument of type string
```

Optional Arguments

```
--add-pool expects an argument of type string
--delete-pool expects an argument of type string
--ip expects an argument of type ip
--weight expects an argument of type gslb-weight
--monitor-ip argument should be one of ip-or-empty
```

Edit a pool

```
lbcli edit-gslb-pool
or
lbcli --action gslb --function edit --section pools
```

Edit a GSLB pool.

Required Arguments

```
--name expects an argument of type string
```

Optional Arguments

```
--add-globalname expects an argument of type string
--add-member expects an argument of type string
--delete-globalname expects an argument of type string
--delete-member expects an argument of type string
--fallback expects an argument of type string
--lb-method argument should be one of wrr, twrr or fogroup
--max-addrs-returned expects an argument of type string
--monitor argument should be one of http, tcp, forced,
external, external_weight or
externaldynamicweight
--monitor-expected-codes expects an argument of type http-code-list
--monitor-hostname expects an argument of type string
--monitor-interval expects an argument of type string
--monitor-parameters expects an argument of type string
--monitor-port expects an argument of type string
--monitor-result expects an argument of type string
```



<code>--monitor-retries</code>	expects an argument of type string
<code>--monitor-return-match</code>	expects an argument of type string
<code>--monitor-script</code>	expects an argument of type string
<code>--monitor-send-string</code>	expects an argument of type string
<code>--monitor-status</code>	argument should be one of up or down
<code>--monitor-timeout</code>	expects an argument of type string
<code>--monitor-url-path</code>	expects an argument of type string
<code>--monitor-use-ssl</code>	expects an argument of type bool

Edit a topology

```
lbcli edit-gslb-topology
or
lbcli --action gslb --function edit --section topologies
```

Edit a GSLB topology.

Required Arguments

<code>--name</code>	expects an argument of type string
---------------------	------------------------------------

Optional Arguments

<code>--add-ips</code>	expects an argument of type string
<code>--delete-ips</code>	expects an argument of type string

Edit HAProxy SSL Associated Termination

```
lbcli edit-haproxy-associated-termination
```

Edit an HAProxy SSL Associated Termination

Required Arguments

<code>--vip</code>	expects an argument of type existing-vip
--------------------	--

Optional Arguments

<code>--certificate</code>	expects an argument of type strict-label
<code>--ca-certificate</code>	argument should be one of strict-label-or-empty
<code>--port</code>	expects an argument of type port
<code>--disablesslv3</code>	expects an argument of type bool
<code>--disabletlsv1</code>	expects an argument of type bool
<code>--disabletlsv1-1</code>	expects an argument of type bool
<code>--disabletlsv1-2</code>	expects an argument of type bool
<code>--disabletlsv1-3</code>	expects an argument of type bool
<code>--ciphers</code>	argument should be one of cipher-list-or-empty
<code>--alpn</code>	argument should be one of alpn-list-or-empty

Modify a health check



```
lbcli edit-healthcheck
```

Modify an existing health check on the cluster. If the `--master` argument is omitted then the `--slave` argument cannot be specified. If `--base64` is yes then the health check body should be base64 encoded. If `--zlib` is set then the health check body should be compressed according to RFC-1950. If both base64 encoding and compression are used then the body should first be compressed then base64 encoded.

Required Arguments

```
--label      expects an argument of type existing-healthcheck
```

Optional Arguments

```
--new-label  expects an argument of type string
--base64     expects an argument of type bool defaults to no
--zlib       expects an argument of type bool defaults to no
--kind       argument should be one of check or gslbcheck
--master     expects an argument of type string
--slave      expects an argument of type string
```

Edit a layer 4 real server

```
lbcli edit-layer4-server
or
lbcli --action edit-rip --layer 4
```

Edit the specified layer 4 real server.

Required Arguments

```
--vip      expects an argument of type existing-vip
--rip      expects an argument of type existing-rip
```

Optional Arguments

```
--ip       expects an argument of type ip
--port     expects an argument of type port
--weight   expects an argument of type rip-weight
--minconns expects an argument of type uint
--maxconns expects an argument of type uint
```

Edit a layer 4 virtual service

```
lbcli edit-layer4-service
or
lbcli --action edit-vip --layer 4
```

Edit the specified layer 4 virtual service.

Required Arguments



<code>--vip</code>	expects an argument of type existing-vip
--------------------	--

Optional Arguments

<code>--autoscale-group</code>	expects an argument of type string
<code>--check-command</code>	expects an argument of type string
<code>--check-database</code>	expects an argument of type string
<code>--check-login</code>	expects an argument of type string
<code>--check-password</code>	expects an argument of type string
<code>--check-port</code>	expects an argument of type port
<code>--check-request</code>	expects an argument of type string
<code>--check-response</code>	expects an argument of type string
<code>--check-secret</code>	expects an argument of type string
<code>--check-service</code>	argument should be one of none, dns, ftp, http, http_proxy, https, imap, imaps, ldap, mysql, nntp, pop, pops, radius, simpletcp, sip or smtp
<code>--check-type</code>	argument should be one of negotiate, connect, ping, external, off, on, 5 or 10
<code>--check-vhost</code>	expects an argument of type string
<code>--email</code>	expects an argument of type string
<code>--email-from</code>	expects an argument of type string
<code>--fallback-ip</code>	expects an argument of type ip
<code>--fallback-local</code>	expects an argument of type bool
<code>--fallback-port</code>	expects an argument of type port
<code>--feedback</code>	argument should be one of agent, http or none
<code>--feedback-port</code>	expects an argument of type port
<code>--forwarding</code>	argument should be one of snat, masq, gate or ipip
<code>--granularity</code>	expects an argument of type cidr
<code>--ip</code>	expects an argument of type ip
<code>--negate-response</code>	expects an argument of type bool
<code>--persist-time</code>	expects an argument of type uint
<code>--persistent</code>	expects an argument of type bool
<code>--ports</code>	argument should be one of port-list-or-asterisk
<code>--protocol</code>	argument should be one of ops, udp, tcp, tcpudp or fwm
<code>--scheduler</code>	argument should be one of wlc, wrd or dh
<code>--send-rip-name-as-host</code>	expects an argument of type bool
<code>--slave-ip</code>	expects an argument of type ip

Edit a layer 7 real server

```
lbcli edit-layer7-server
or
lbcli --action edit-rip --layer 7
```

Edit the specified layer 7 real server.

Required Arguments

<code>--vip</code>	expects an argument of type existing-vip
<code>--rip</code>	expects an argument of type existing-rip

Optional Arguments



<code>--ip</code>	expects an argument of type ip
<code>--port</code>	expects an argument of type port
<code>--weight</code>	expects an argument of type rip-weight
<code>--minconns</code>	expects an argument of type uint
<code>--maxconns</code>	expects an argument of type uint
<code>--encrypted</code>	expects an argument of type bool
<code>--cert</code>	argument should be one of existing-certificate-or-empty
<code>--cacert</code>	argument should be one of existing-cacert-family-or-empty
<code>--redir</code>	expects an argument of type string
<code>--redir-enabled</code>	expects an argument of type bool

Edit a layer 7 virtual service

```
lbcli edit-layer7-service
or
lbcli --action edit-vip --layer 7
```

Edit the specified layer 7 virtual service.

Required Arguments

<code>--vip</code>	expects an argument of type existing-vip
--------------------	--

Optional Arguments

<code>--appsession-cookie</code>	expects an argument of type string
<code>--as-port</code>	expects an argument of type port
<code>--autoscale-group</code>	expects an argument of type string
<code>--backend-encryption</code>	expects an argument of type bool
<code>--backend-only</code>	expects an argument of type bool
<code>--ca-certificate</code>	argument should be one of existing-cacert-family-or-empty
<code>--certificate</code>	expects an argument of type existing-certificate
<code>--check-host</code>	expects an argument of type string
<code>--check-negate-response</code>	expects an argument of type bool
<code>--check-password</code>	expects an argument of type string
<code>--check-port</code>	argument should be one of port-or-empty
<code>--check-receive</code>	expects an argument of type string
<code>--check-request</code>	expects an argument of type string
<code>--check-type</code>	argument should be one of connect, external, negotiate_http, negotiate_https, negotiate_http_head, negotiate_https_head, negotiate_http_options, negotiate_https_options, mysql or none
<code>--check-username</code>	expects an argument of type string
<code>--ciphers</code>	expects an argument of type cipher-list
<code>--clear-stick-drain</code>	expects an argument of type bool
<code>--compression</code>	expects an argument of type bool
<code>--cookie-attr-domain</code>	expects an argument of type space-seperated-cookie-domains
<code>--cookie-attr-httponly</code>	expects an argument of type bool
<code>--cookie-attr-secure</code>	expects an argument of type bool
<code>--cookie-maxidle</code>	expects an argument of type haproxy-interval
<code>--cookie-maxlife</code>	expects an argument of type haproxy-interval



<code>--cookie-name</code>	expects an argument of type string
<code>--default-ciphers</code>	argument should be one of cipher-list-or-empty
<code>--disable-ssl-v3</code>	expects an argument of type bool
<code>--disable-tls-v1</code>	expects an argument of type bool
<code>--disable-tls-v1-1</code>	expects an argument of type bool
<code>--disable-tls-v1-2</code>	expects an argument of type bool
<code>--disable-tls-v1-3</code>	expects an argument of type bool
<code>--enable-hsts</code>	expects an argument of type bool
<code>--encrypt-all-backends</code>	expects an argument of type bool
<code>--external-check-script</code>	expects an argument of type existing-healthcheck
<code>--fallback-disabled</code>	expects an argument of type bool
<code>--fallback-encrypt</code>	expects an argument of type bool
<code>--fallback-ip</code>	expects an argument of type ip
<code>--fallback-persist</code>	expects an argument of type bool
<code>--fallback-port</code>	expects an argument of type port
<code>--feedback-method</code>	argument should be one of agent or none
<code>--feedback-port</code>	expects an argument of type port
<code>--force-to-https</code>	expects an argument of type bool
<code>--force-to-https-port</code>	argument should be one of port-or-empty
<code>--forward-for</code>	expects an argument of type bool
<code>--hsts-month</code>	expects an argument of type uint
<code>--http-pipeline</code>	argument should be one of http_keep_alive, http_close, http_server_close or http_force_close
<code>--http-pretend-keepalive</code>	expects an argument of type bool
<code>--http-request</code>	expects an argument of type bool
<code>--invalid-http</code>	expects an argument of type bool
<code>--ip</code>	expects an argument of type ip
<code>--maxconn</code>	expects an argument of type uint
<code>--mode</code>	argument should be one of other_tcp, tcp, ip, http or http2
<code>--no-write</code>	expects an argument of type bool
<code>--persist-table-size</code>	expects an argument of type uint
<code>--persist-time</code>	expects an argument of type uint
<code>--persistence</code>	argument should be one of none, ip, tcp, sslsesid, http, appsession, rdp-session, rdp-cookie, http_ip, waf, last, xff or fallback_persist
<code>--ports</code>	expects an argument of type port-list
<code>--proxy-bind</code>	expects an argument of type string
<code>--redirect-code</code>	expects an argument of type http-300-code
<code>--redispatch</code>	expects an argument of type bool
<code>--reuse-idle</code>	expects an argument of type bool
<code>--scheduler</code>	argument should be one of roundrobin, leastconn or first
<code>--send-proxy</code>	argument should be one of none, v1, v2, v2_ssl or v2_ssl_cn defaults to none
<code>--send-rip-name-as-host</code>	expects an argument of type bool
<code>--slave-ip</code>	expects an argument of type ip
<code>--source-address</code>	expects an argument of type ip
<code>--stunnel-source</code>	expects an argument of type string
<code>--stunnel-proxy</code>	expects an argument of type bool
<code>--tcp-keep-alive</code>	expects an argument of type bool
<code>--timeout</code>	expects an argument of type bool
<code>--timeout-client</code>	expects an argument of type haproxy-interval
<code>--timeout-server</code>	expects an argument of type haproxy-interval
<code>--tproxy</code>	expects an argument of type bool

<code>--tunneltimeout</code>	expects an argument of type haproxy-interval
<code>--use-content-inspection</code>	expects an argument of type bool
<code>--verify-client-certificate</code>	argument should be one of required, optional or none
<code>--xff-ip-pos</code>	expects an argument of type int

Edit a SSL termination

```
lbcli edit-termination
or
lbcli --action termination --function edit --type stunnel
```

Edit the specified SSL termination.

Required Arguments

<code>--vip</code>	expects an argument of type existing-termination
<code>--proxy-bind</code>	expects an argument of type bool defaults to false

Optional Arguments

<code>--port</code>	expects an argument of type port
<code>--associated-to</code>	expects an argument of type existing-l7-vip
<code>--proxy-bind</code>	expects an argument of type existing-l7-vip
<code>--disablesslv2</code>	expects an argument of type bool
<code>--disablesslv3</code>	expects an argument of type bool
<code>--disabletlsv1</code>	expects an argument of type bool
<code>--stunneldnsdelay</code>	expects an argument of type bool
<code>--stunnelproxy</code>	expects an argument of type bool
<code>--servercipherorder</code>	expects an argument of type bool
<code>--emptyfragments</code>	expects an argument of type bool
<code>--stunnelrenegotiation</code>	expects an argument of type bool
<code>--stunneltimetoclose</code>	expects an argument of type uint
<code>--sslcert</code>	expects an argument of type existing-certificate
<code>--disabletlsv1-1</code>	expects an argument of type bool
<code>--disabletlsv1-2</code>	expects an argument of type bool
<code>--disabletlsv1-3</code>	expects an argument of type bool
<code>--sslmode</code>	argument should be one of high, fips or compatable
<code>--source</code>	expects an argument of type ip
<code>--slave-source</code>	expects an argument of type ip
<code>--ciphers</code>	expects an argument of type cipher-list
<code>--ip</code>	expects an argument of type ip
<code>--slave-ip</code>	expects an argument of type ip
<code>--backend-ip</code>	expects an argument of type ip
<code>--slave-backend-ip</code>	expects an argument of type ip
<code>--backend-port</code>	expects an argument of type port

Edit a WAF

```
lbcli edit-waf
```

Edit the specified Web Application Firewall.

Required Arguments

<code>--waf</code>	expects an argument of type existing-waf
--------------------	--

Optional Arguments

<code>--in-anom-score</code>	expects an argument of type uint
<code>--out-anom-score</code>	expects an argument of type uint
<code>--req-data</code>	expects an argument of type bool
<code>--resp-data</code>	expects an argument of type bool
<code>--audit</code>	expects an argument of type bool
<code>--dlogin</code>	expects an argument of type bool
<code>--dlogin-mode</code>	argument should be one of static, openid_google or ldap
<code>--dlogin-google-clientid</code>	expects an argument of type string
<code>--dlogin-google-clientsecret</code>	expects an argument of type string
<code>--dlogin-google-redirect-uri</code>	expects an argument of type string
<code>--dlogin-google-passphrase</code>	expects an argument of type string
<code>--dlogin-google-allowed-domain</code>	expects an argument of type string
<code>--rule-engine</code>	expects an argument of type bool
<code>--proxytimeout</code>	expects an argument of type uint
<code>--dlogin-location</code>	expects an argument of type string
<code>--dlogin-static-username</code>	expects an argument of type string
<code>--dlogin-static-password</code>	expects an argument of type string
<code>--ldap-server</code>	expects an argument of type string
<code>--ldap-server-port</code>	expects an argument of type port
<code>--search-base</code>	expects an argument of type string
<code>--ldap-attribute</code>	expects an argument of type string
<code>--ldap-group</code>	expects an argument of type string
<code>--bind-user</code>	expects an argument of type string
<code>--bind-password</code>	expects an argument of type string
<code>--disable-waf</code>	expects an argument of type bool
<code>--cacheaccel</code>	expects an argument of type bool
<code>--cache-nocache-files</code>	expects an argument of type string
<code>--cache-force-cache</code>	expects an argument of type bool
<code>--cache-object-size</code>	expects an argument of type uint

Enable the API endpoint

```
lbcli enable-api
or
lbcli --action api --function enable
```

Enable the API endpoint on the appliance. The apikey must be the unencoded value of the base64 string passed in the X-LB-APIKEY header of all requests.

Required Arguments

<code>--username</code>	expects an argument of type string defaults to loadbalancer
<code>--password</code>	expects an argument of type string defaults to loadbalancer

Optional Arguments

<code>--apikey</code>	expects an argument of type string
-----------------------	------------------------------------



Enable Floating IP.

```
lbcli enable-floating-ip
```

Make it so that the given floating IP can be made active on this appliance.

Remove and add a floating IP address

```
lbcli fix-floating-ip
```

Remove and re-add a floating IP in order to ensure it is up after altering the kernel state.

Required Arguments

```
--ip expects an argument of type ip
```

Remove all ACLs for a virtual service

```
lbcli flush-acls
```

Remove all ACLS for the specified layer 7 virtual service.

Required Arguments

```
--vip expects an argument of type existing-l7-vip
```

Remove all static IP addresses

```
lbcli flush-static-ips  
or  
lbcli --action address --function flush
```

Remove all static IP addresses from an appliance.

Optional Arguments

```
--interface expects an argument of type existing-interface
```

Remove all static routes

```
lbcli flush-static-routes  
or  
lbcli --action route --function static --type flush
```

Remove all static routes from an appliance.

Show the properties of the API endpoint

```
lbcli get-api  
or
```



```
lbcli --action api --function get
```

Show the username, password and unencoded X-LB_APIKEY header value expected by the API endpoint.

Show the default route

```
lbcli get-default-route  
or  
lbcli --action route --function default --type get
```

Show the default route of the default table for this appliance.

Show the DNS servers

```
lbcli get-dns  
or  
lbcli --action dns --function get
```

Show the Domain Name Servers for this appliance.

Show the firewall settings

```
lbcli get-firewall  
or  
lbcli --action local-configuration --get firewall
```

Show the settings for the firewall for this appliance.

Show the graphing settings

```
lbcli get-graphing  
or  
lbcli --action local-configuration --get graphing
```

Show the graphing settings for this appliance.

Show the GSLB reports

```
lbcli get-gslb-reports  
or  
lbcli --action gslb --function get --section reports
```

Show the GSLB reports for this appliance.

Required Arguments

```
--report argument should be one of get_generic_state or get_ppdns_state
```

Show the hostname



```
lbcli get-hostname
or
lbcli --action hostname --function get
```

Show the host and domain name for this appliance.

Show the interface offload settings

```
lbcli get-interface-offload
or
lbcli --action local-configuration --get interface_offload
```

Show the interface offload settings for this appliance.

Show the management gateway

```
lbcli get-management-gateway
or
lbcli --action local-configuration --get management_gateway
```

Show the details of the routing rule that traffic from this appliance's WebUI will return via.

Show the interface MTU settings

```
lbcli get-mtu
or
lbcli --action mtu --function get
```

Show the MTU settings for the interfaces on this appliance.

Show the NTP servers

```
lbcli get-ntp
or
lbcli --action local-configuration --get ntp
```

Show the Network Time Protocol servers this appliance will use.

Show the HTTP proxy server settings

```
lbcli get-proxy
or
lbcli --action local-configuration --get proxy
```

Show the HTTP proxy server the WebUI on this appliance will use.

Show the security settings

```
lbcli get-security
or
```



```
lbcli --action local-configuration --get security
```

Show the security settings for this appliance.

Show the email smarthost

```
lbcli get-smarthost  
or  
lbcli --action local-configuration --get smarthost
```

Show the email smarthost this appliance will use to send emails.

Show the email smarthost

```
lbcli get-smtp  
or  
lbcli --action local-configuration --get smtp
```

Show the email smarthost this appliance will use to send emails.

Show the syslog settings

```
lbcli get-syslog  
or  
lbcli --action local-configuration --get syslog
```

Show the syslog settings for this appliance.

Show the current date and time

```
lbcli get-timedate  
or  
lbcli --action local-configuration --get timedate
```

Show the current date and time for this appliance.

Show the URL of the updates server

```
lbcli get-updates  
or  
lbcli --action local-configuration --get updates
```

Show the URL of the updates server this appliance will use to fetch online updates.

Create a cluster

```
lbcli ha_create  
or  
lbcli create-ha
```



Pair this appliance with another to create a high availability cluster. This appliance will become the primary server.

Required Arguments

```
--local-ip      expects an argument of type ip
--peer-ip       expects an argument of type string
--peer-password expects an argument of type string
```

Halt a real server

```
lbcli halt
```

Close all connections to the specified real server and forbid and new connections.

Required Arguments

```
--vip expects an argument of type existing-vip
--rip expects an argument of type existing-rip
```

Clear all HAProxy stick tables

```
lbcli haproxy-clear-stick
```

Clear all HAProxy stick tables.

Install a licence

```
lbcli install-licence
```

Install a key to licence this appliance.

Required Arguments

```
--key expects an argument of type string
```

List ACL rules

```
lbcli list-acls
or
lbcli --action acl --function list
```

List all ACL rules for the specified layer 7 virtual service.

Required Arguments

```
--vip expects an argument of type existing-l7-vip
```

List CA Certificates



```
lbcli list-ca-certificates
```

List the CA Certificates in a Family

Required Arguments

```
--family expects an argument of type existing-cacert-family
```

List SSL certificates

```
lbcli list-certificates  
or  
lbcli --action termination --function list --type certificate
```

List all SSL certificates on this cluster.

List floating IP addresses

```
lbcli list-floating-ips  
or  
lbcli --action list --function floatingip
```

List all floating IPs on this cluster.

List global names

```
lbcli list-gslb-globalnames  
or  
lbcli --action gslb --function list --section globalnames
```

List all GSLB global names.

List members

```
lbcli list-gslb-members  
or  
lbcli --action gslb --function list --section members
```

List all GSLB members.

List pools

```
lbcli list-gslb-pools  
or  
lbcli --action gslb --function list --section pools
```

List all GSLB pools.

List topologies



```
lbcli list-gslb-topologies
or
lbcli --action gslb --function list --section topologies
```

List all GSLB topologies.

List header rules

```
lbcli list-headers
or
lbcli --action headers --function list
```

List all header rules for the specified layer 7 virtual service.

Required Arguments

```
--vip expects an argument of type existing-l7-vip
```

List health checks

```
lbcli list-healthchecks
```

List existing health checks on the cluster.

List PBR rules

```
lbcli list-pbr
or
lbcli --action pbr --function get
lbcli get-pbr
```

List Policy Based Routing rules for this appliance.

Optional Arguments

```
--ip expects an argument of type ip
--gateway expects an argument of type ip
```

List static IP addresses

```
lbcli list-static-ips
or
lbcli --action address --function get
```

List static IPs configured on the specified interface.

Required Arguments

```
--interface expects an argument of type existing-interface
```



List static routes

```
lbcli list-static-routes  
or  
lbcli --action route --function static --type get
```

List static routes configured on this appliance.

List WAFs

```
lbcli list-waf
```

List Web Application Firewalls configured on this appliance.

Lockdown access to the WebUI

```
lbcli lockdown
```

Lockdown the WebUI on this appliance so that it only responds to clients on the specified subnet.

Required Arguments

```
--enabled expects an argument of type bool
```

Optional Arguments

```
--network expects an argument of type ip-and-cidr
```

Report the node statuses

```
lbcli nodestatus
```

Report the status of the nodes in this cluster as this appliance understands them.

Bring a real server online

```
lbcli online
```

Bring a real server that is halted or drained back online by allowing it to receive connections once more.

Required Arguments

```
--vip expects an argument of type existing-vip  
--rip expects an argument of type existing-rip
```

Reboot appliance

```
lbcli reboot  
or
```



```
lbcli --action power --function restart
```

Reboot this appliance.

Regenerate default certificate

```
lbcli regenerate-default-certificate  
or  
lbcli --action termination --function regenerate_local --type certificate
```

Regenerate the default self-signed certificate used by the webui and SSL-terminations.

Reload WebUI

```
lbcli reload-apache  
or  
lbcli reload-webui
```

Reload the WebUI service on this appliance.

Reload GSLB

```
lbcli reload-gslb
```

Reload the GSLB service on this appliance.

Reload HAProxy

```
lbcli reload-haproxy
```

Reload the HAProxy service on this appliance.

Reload Heartbeat

```
lbcli reload-heartbeat
```

Reload the Heartbeat service on this appliance.

Reload LDirectord

```
lbcli reload-ldirectord
```

Reload the LDirectord service on this appliance.

Reload STunnel

```
lbcli reload-stunnel
```

Reload the STunnel service on this appliance.



Reload Syslog

```
lbcli reload-syslog
```

Reload the Syslog service on this appliance.

Reload WAF

```
lbcli reload-waf
```

Reload the WAF service on this appliance.

Break a cluster

```
lbcli reset-cluster-config  
or  
lbcli --action reset --function cluster
```

Break a high-availability pair.

Restart Autoscaled

```
lbcli restart-autoscaling
```

Reload the Autoscaled service on this appliance.

Restart AZHA

```
lbcli restart-azha
```

Reload the AZHA service on this appliance.

Restart Collectd

```
lbcli restart-collectd
```

Reload the Collectd service on this appliance.

Restart Firewall

```
lbcli restart-firewall
```

Reload the Firewall service on this appliance.

Restart GSLB

```
lbcli restart-gslb
```

Restart the GSLB service on this appliance.



Restart HAProxy

```
lbcli restart-haproxy
```

Reload the HAProxy service on this appliance.

Restart Heartbeat

```
lbcli restart-heartbeat
```

Reload the Heartbeat service on this appliance.

Restart LDirectord

```
lbcli restart-ldirectord
```

Reload the LDirectord service on this appliance.

Restart Pound

```
lbcli restart-pound
```

Reload the Pound service on this appliance.

Restart SNMP

```
lbcli restart-snmp
```

Reload the SNMP service on this appliance.

Restart STunnel

```
lbcli restart-stunnel
```

Reload the STunnel service on this appliance.

Restart Syslog

```
lbcli restart-syslog
```

Reload the Syslog service on this appliance.

Restart WAF

```
lbcli restart-waf
```

Reload the WAF service on this appliance.



Restore factory defaults

```
lbcli restore
```

Restore this appliance to factory defaults.

Required Arguments

```
--yes-i-am-sure expects an argument of type bool
```

Set default route

```
lbcli set-default-route  
or  
lbcli --action route --function default --type set
```

Set the default gateway for this appliance.

Required Arguments

```
--interface argument should be one of existing-interface-or-auto defaults  
              to auto  
--gateway    expects an argument of type ip
```

Set DNS servers

```
lbcli set-dns  
or  
lbcli --action dns --function set
```

Set the Domain Name Servers for this appliance.

Optional Arguments

```
--dns0 argument should be one of ip-or-empty  
--dns1 argument should be one of ip-or-empty  
--dns2 argument should be one of ip-or-empty
```

Set firewall properties

```
lbcli set-firewall  
or  
lbcli --action local-configuration --set firewall
```

Set the properties for the firewall for this appliance.

Required Arguments

```
--conntrack-size expects an argument of type uint
```



Set graphing properties

```
lbcli set-graphing
or
lbcli --action local-configuration --set graphing
```

Set the graphing properties for this appliance.

Optional Arguments

```
--layer4      expects an argument of type bool
--layer7      expects an argument of type bool
--interfaces  expects an argument of type bool
--interfaces  expects an argument of type bool
--load        expects an argument of type bool
--memory      expects an argument of type bool
--disk        expects an argument of type bool
--interval    expects an argument of type uint
--timeout     expects an argument of type uint
```

Set hostname

```
lbcli set-hostname
or
lbcli --action hostname --function set
```

Set the host and domain name for this appliance.

Required Arguments

```
--hostname  expects an argument of type string
--domain    expects an argument of type string defaults to localdomain
```

Set interface offload

```
lbcli set-interface-offload
or
lbcli --action local-configuration --set interface_offload
```

Enable or disable TCP offload for this appliance.

Required Arguments

```
--hardware-offload  expects an argument of type bool
```

Set the management gateway

```
lbcli set-management-gateway
or
lbcli --action local-configuration --set management_gateway
```

Set the routing rule which traffic from the WeBUI will follow for this appliance.



Required Arguments

```
--iface-ip  expects an argument of type existing-static-ip
--gateway   expects an argument of type ip
```

Set the MTU

```
lbcli set-mtu
or
lbcli --action mtu --function set
```

Set the MTU for the specified interface.

Required Arguments

```
--interface expects an argument of type existing-interface
--mtu        expects an argument of type mtu
```

Set NTP server

```
lbcli set-ntp
or
lbcli --action local-configuration --set ntp
```

Set the Network Time Protocol servers used by this appliance.

Set HTTP proxy

```
lbcli set-proxy
or
lbcli --action local-configuration --set proxy
```

Set the HTTP proxy used by the WebUI of this appliance.

Optional Arguments

```
--ip          argument should be one of hostname-or-ip
--port        expects an argument of type port
--username    expects an argument of type string
--password    expects an argument of type string
```

Set security settings

```
lbcli set-security
or
lbcli --action local-configuration --set security
```

Set the security settings for this appliance.

Optional Arguments



```
--security-mode    argument should be one of custom, secure or secure-perm
--rootaccess       expects an argument of type bool
--sshpasword       expects an argument of type bool
--httpsonly        expects an argument of type bool
--httpsport        expects an argument of type bool
--certificate       expects an argument of type existing-certificate
--httpsciphers      expects an argument of type cipher-list
```

Set the email smarthost

```
lbcli set-smarthost
or
lbcli --action local-configuration --set smarthost
```

Set the email smarthost this appliance will use to send emails.

Optional Arguments

```
--smtp-relay  expects an argument of type string
```

Set SNMP properties

```
lbcli set-snmp
or
lbcli --action local-configuration --set snmp
```

Set the properties of the SNMP service on this appliance.

Optional Arguments

```
--community      expects an argument of type string
--location        expects an argument of type string
--contact         expects an argument of type string
--user            expects an argument of type string
--auth-method     expects an argument of type string
--auth           expects an argument of type string
--priv-method     expects an argument of type string
--priv           expects an argument of type string
--snmpv3-enable   expects an argument of type string
--snmpv2-enable   expects an argument of type string
```

Set syslog properties

```
lbcli set-syslog
or
lbcli --action local-configuration --set syslog
```

Set the properties of the syslog service on this appliance.

Optional Arguments

```
--interval      expects an argument of type uint
```



```
--burst      expects an argument of type uint
--destination expects an argument of type string
--ip         expects an argument of type ip
--port       expects an argument of type port
--protocol   argument should be one of tcp or udp
--template   expects an argument of type string
```

Set time and date

```
lbcli set-timedate
or
lbcli --action local-configuration --set timedate
```

Set the time and date of this appliance.

Required Arguments

```
--year      expects an argument of type uint
--month      expects an argument of type month
--day        expects an argument of type day
--hour       expects an argument of type hour
--minute     expects an argument of type minute
```

Enable or disable update check

```
lbcli set-updates
or
lbcli --action local-configuration --set updates
```

Enable or disable the update check on this appliance.

Optional Arguments

```
--auto-check-for-updates expects an argument of type bool
--update-server           expects an argument of type string
```

Show the sections of a CSR

```
lbcli show-csr
```

Show the sections of a CSR

Required Arguments

```
--certificate expects an argument of type existing-certificate
```

Show configuration.

```
lbcli show-full-config
or
lbcli --action list --function dumpconfig
```



Show configuration for this appliance.

Show configuration for LDirectord.

```
lbcli show-layer4-advanced
or
lbcli --action list --function advanced --layer 4
```

Show configuration for LDirectord for the cluster.

Show configuration for Layer 4 virtual services.

```
lbcli show-layer4-services
or
lbcli --action list --function virtual --layer 4
```

Show the configuration for Layer 4 virtual services on this cluster.

Show configuration for HAProxy.

```
lbcli show-layer7-advanced
or
lbcli --action list --function advanced --layer 7
```

Show configuration for HAProxy for the cluster.

Show configuration for Layer 7 virtual services.

```
lbcli show-layer7-services
or
lbcli --action list --function virtual --layer 7
```

Show the configuration for Layer 7 virtual services on this cluster.

Show a Certificate Revocation List

```
lbcli show-revocation-list
```

Show Certificate Revocation List for a Family

Required Arguments

```
--family expects an argument of type existing-cacert-family
```

Shutdown appliance.

```
lbcli shutdown
or
lbcli --action power --function shutdown
```



Shutdown this appliance.

Show GSLB status.

```
lbcli status-gslb
```

Show the status of the GSLB service for this appliance.

Create a technical support download.

```
lbcli support-download
```

Create a technical support download for this appliance.

Get uptime.

```
lbcli uptime
```

Get the uptime for this appliance.

Running lbcli from a remote Linux Host

These commands can also be run from a remote Linux host. This example halts VIP1/RIP1:

```
ssh root@192.168.111.42 "lbcli --action halt --vip VIP1 --rip RIP1"
```

Running lbcli from a remote Windows Host

These commands can also be run from a remote Windows host. This example halts VIP1/RIP1:

```
plink -pw loadbalancer root@192.168.111.42 "lbcli --action halt --vip VIP1 --rip RIP1"
```

Notes

1. PuTTY must be installed to use the *plink* command.
2. The password for the "root" user is set during the Network Setup Wizard.
3. 192.168.111.42 is the IP address of the load balancer.

Application Programming Interface (API)

Enabling the API

To enable the API, run the following command:

```
lbcli --action api --function enable --username (username) --password (password) --apikey (apikey)
```

e.g. The following command sets the API username to "loadbalancer", the password to "loadbalancer" and the apikey to "Aplk3y2345118".



```
lbcli --action api --function enable --username loadbalancer --password loadbalancer --apikey
ApIk3y2345118
```



Note

To automatically generate a 32 character random key, omit the --apikey option.

Once enabled, the API enables all CLI commands to be executed using HTTP POST requests.

When making API calls, the base64 encoded version of the apikey must be specified.

To obtain the base64 encoded version of the apikey, the following command can be used:

```
echo '<apikey>' | base64
```

for example, if the apikey is set to "ApIk3y2345118" as in the example above, the following command can be used:

```
echo 'ApIk3y2345118' | base64
QXBJazN5MjM0NTExOAo=
```

In this case **QXBJazN5MjM0NTExOAo=** would be used in API requests. Please refer to the examples in [Sending API Requests](#).

API Endpoint

API calls must be posted to the following URL on the load balancer:

```
https://<appliance IP address>:9443/api/v2/
```



Note

If the WebUI has been configured to listen on a different port, modify the URL accordingly. For more details on setting the port, please refer to [Service Socket Addresses](#).

Creating the JSON Request

To illustrate how JSON requests are formed, the following examples show the CLI command and the corresponding JSON request in each case.

Example 1 - Halt a Real Server

This example shows how RIP1 of VIP1 can be halted.

lbcli command:

```
lbcli --action halt --vip VIP1 --rip RIP1
```

JSON equivalent:



```
{
  "lbcli": [{
    "action": "halt",
    "vip": "VIP1",
    "rip": "RIP1"
  }]
}
```

Example 2 - Add a Layer 7 VIP

This example shows how to add a Layer 7 HTTP mode VIP.

lbcli command:

```
lbcli --action add-vip --layer 7 --vip VIP1 --ip 192.168.1.1 --ports 80 --mode http
```

JSON equivalent:

```
{
  "lbcli": [{
    "action": "add-vip",
    "layer": "7",
    "vip": "VIP1",
    "ip": "192.168.1.1",
    "ports": "80",
    "mode": "http"
  }]
}
```

Example 3 - Restart HAProxy

This example shows how to restart HAProxy.

lbcli command:

```
lbcli --action restart-haproxy
```

JSON equivalent:

```
{
  "lbcli": [{
    "action": "restart-haproxy"
  }]
}
```

JSON Syntax Validation

The website <https://jsonlint.com/> is a useful resource for JSON syntax validation. Paste the JSON into the

window provided and click **Validate JSON**.

Sending API Requests

Using Postman

Postman is a great tool for testing and verifying API functionality. For more details on using Postman, please refer to [our blog](#).

Using script/code

Example 1 - Using the Linux curl command

```
curl -u loadbalancer:loadbalancer -X POST https://192.168.110.150:9443/api/v2/ -H "X-LB-APIKEY: QXBJazN5MjM0NTExOAAo=" -H "Content-Type:application/json" -d '{"lbcli":[{"action":"dns","function":"get"}]}' -k
```

Example 2 - Using Microsoft PowerShell

```
# Configure Variables
$user = "loadbalancer" # Appliance API Username
$pass = "loadbalancer" # Appliance API Password
$apikey = "ApIk3y2345118" # Appliance Non base64 encoded APIKEY
$ip = "192.168.111.220" # Appliance Management IP address
$jsonfile = "c:\api-json\add-vip.json" # Local path to JSON configuration file

# Disable certificate verification checks to allow for the self signed WebUI certificate on the load balancer
if (-not ([System.Management.Automation.PSTypeName]'TrustAllCertsPolicy').Type)
{
    add-type @"
        using System.Net;
        using System.Security.Cryptography.X509Certificates;
        public class TrustAllCertsPolicy : ICertificatePolicy {
            public bool CheckValidationResult(
                ServicePoint srvPoint, X509Certificate certificate,
                WebRequest request, int certificateProblem) {
                return true;
            }
        }
    "@
}

[System.Net.ServicePointManager]::CertificatePolicy = New-Object TrustAllCertsPolicy

# Base64 encode the apkey
$bytes_api = [System.Text.Encoding]::ASCII.GetBytes("$apikey")
$base64_api = [System.Convert]::ToBase64String($bytes_api)
$APIAuthValue = "${base64_api}"

# Set the auth headers
$headers = @"Authorization"=$basicAuthValue; "X-LB-APIKEY"=$APIAuthValue @"

# Fetch the json configuration file
$json = Get-Content $jsonfile -Raw

# Send the API call
```



```
Invoke-WebRequest -Uri ("https://{ip}:9443/api/v2/") -Method Post -Body $json -ContentType
"application/json" -Headers $headers
```

Note

Change the variable declarations to suit your environment.

The add-vip.json file referred to in the above example:

```
{
  "lbcli": [{
    "action": "add-vip",
    "layer": "7",
    "vip": "VIP1",
    "ip": "192.168.111.228",
    "ports": "80",
    "mode": "http"
  }]
}
```

Note

For more examples of making API call using other languages, please refer to our blog [How to automate load balancer deployments, Part 2!](#).

Using ipvsadm to configure Layer 4 Services

For layer 4 services, the ipvsadm command can be used. Several examples are provided below.

Add a TCP based Virtual Service & use weighted round robin scheduling:

```
ipvsadm -A -t 192.168.65.192:80 -s wrr
```

Add a TCP based Real Server in DR mode:

```
ipvsadm -a -t 192.168.65.192:80 -g -r 192.168.70.196:80
```

Add a TCP based Real Server in NAT mode:

```
ipvsadm -a -t 192.168.65.192:80 -m -r 192.168.70.196:80
```

Add a UDP based Virtual Service & use weighted least connection scheduling:

```
ipvsadm -A -u 192.168.65.192:80 -s wlc
```

Add a UDP based Real Server in DR mode:

```
ipvsadm -a -u 192.168.65.192:80 -g -r 192.168.70.196:80
```



Delete a TCP based Virtual Service:

```
ipvsadm -D -t 192.168.65.180:80
```

Delete a TCP based Real Server:

```
ipvsadm -d -t 192.168.65.122:80 -r 192.168.70.134:80
```

View the current running config:

```
ipvsadm -ln
```

Command output:

```
IP Virtual Service version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
-> RemoteAddress:Port Forward Weight ActiveConn InActConn
TCP 192.168.65.120:80 rr
-> 192.168.70.130:80 Route 1 0 0
-> 192.168.70.131:80 Route 1 0 0
TCP 192.168.65.122:80 rr
-> 192.168.70.132:80 Mass 1 0 0
-> 192.168.70.133:80 Mass 1 0 0
```

Note

Please note that since these changes are being made directly to the running configuration, the services that are displayed in the System Overview will no longer match the running configuration when ipvsadm/socat commands are used. Using the **lbcli** command or the API does not have this disadvantage since the System Overview will show the correct VIP and RIP status.

Using Linux socket commands to configure Layer 7 Services

For layer 7 HAProxy VIPs, the socat socket command can be used as shown in the examples below.

To take a server offline:

```
echo "disable server VIP_Name/RIP_Name" | socat unix-connect:/var/run/haproxy.stat stdio
```

To bring a server online:

```
echo "enable server VIP_Name/RIP_Name" | socat unix-connect:/var/run/haproxy.stat stdio
```

To set the weight of a Real Server:



```
echo "set weight VIP_Name/RIP_Name 0" | socat unix-connect:/var/run/haproxy.stat stdio
```

To view HAProxy's running configuration:

```
echo "show info" | socat unix-connect:/var/run/haproxy.stat stdio
```

To clear HAProxy's statistics:

```
echo "clear counters all" | socat unix-connect:/var/run/haproxy.stat stdio
```



Note

Other Linux Socket command examples can be found [here](#).



Note

Since these changes are being made directly to the running configuration, the services that are displayed in the System Overview will no longer match the running configuration when `ipvsadm/socat` commands are used. Using the **lbcli** command or the API does not have this disadvantage since the System Overview will show the correct VIP and RIP status.

Chapter 7 - Web Application Firewall (WAF)

Introduction

A Web Application Firewall (WAF) filters, monitors, and blocks HTTP traffic to and from a web application.

The load balancer includes a built-in WAF. It can be deployed in front of a web application to provide an additional layer of security, where required. It is based on the free and open-source ModSecurity WAF engine and includes the [OWASP ModSecurity Core Rule Set \(CRS\)](#) by default. The CRS is a set of generic attack detection rules. It aims to protect web applications from a wide range of attacks, including the OWASP Top 10, while keeping false positives (false alerts) to a minimum.

The OWASP Top 10 represents a broad consensus about the most critical security risks to web applications. These risks are broken down into ten categories, as shown in the table below:

Category	Description
A01 - Broken Access Control	Access control failures typically lead to unauthorized information disclosure, modification, or destruction of all data or performing a business function outside the user's limits.
A02 - Cryptographic Failures	Previously known as Sensitive Data Exposure , the focus is on failures related to cryptography (or lack thereof). Which often lead to exposure of sensitive data.
A03 - Injection	An application is vulnerable to injection attack when, for example, user-supplied data is not validated, filtered, or sanitized by the application.
A04 - Insecure Design	A new category which focuses on risks related to design and architectural flaws, with a call for more use of threat modeling, secure design patterns, and reference architectures.
A05 - Security Misconfiguration	The application might be vulnerable if the application is, for example, missing appropriate security hardening across any part of the application stack.
A06 - Vulnerable and Outdated Components	You are likely vulnerable, for example, if you do not know the versions of all components you use (both client-side and server-side), including nested dependencies.
A07 - Identification and Authentication Failures	Previously known as Broken Authentication , confirmation of the user's identity, authentication, and session management is critical to protect against authentication-related attacks.
A08 - Software and Data Integrity Failures	A new category which focuses on making assumptions related to software updates, critical data, and CI/CD pipelines without verifying integrity.
A09 - Security Logging and Monitoring Failures	Detecting and responding to breaches is critical. This category is to help detect, escalate, and respond to active breaches.
A10 - Server-Side Request Forgery (SSRF)	SSRF flaws occur whenever a web application is fetching a remote resource without validating the user-supplied URL.

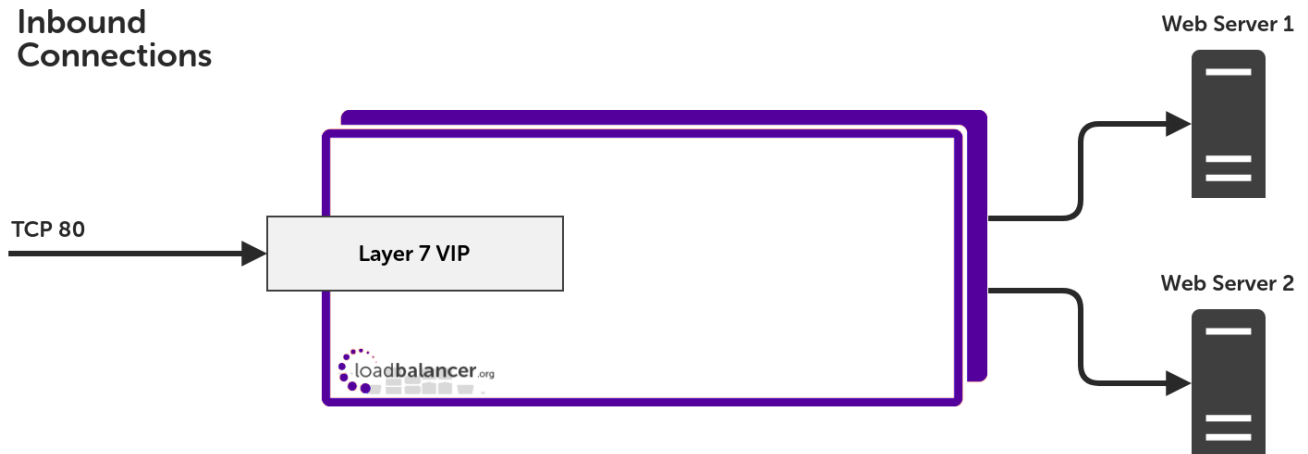
More details can be found at the [OWASP Top 10 website](#).



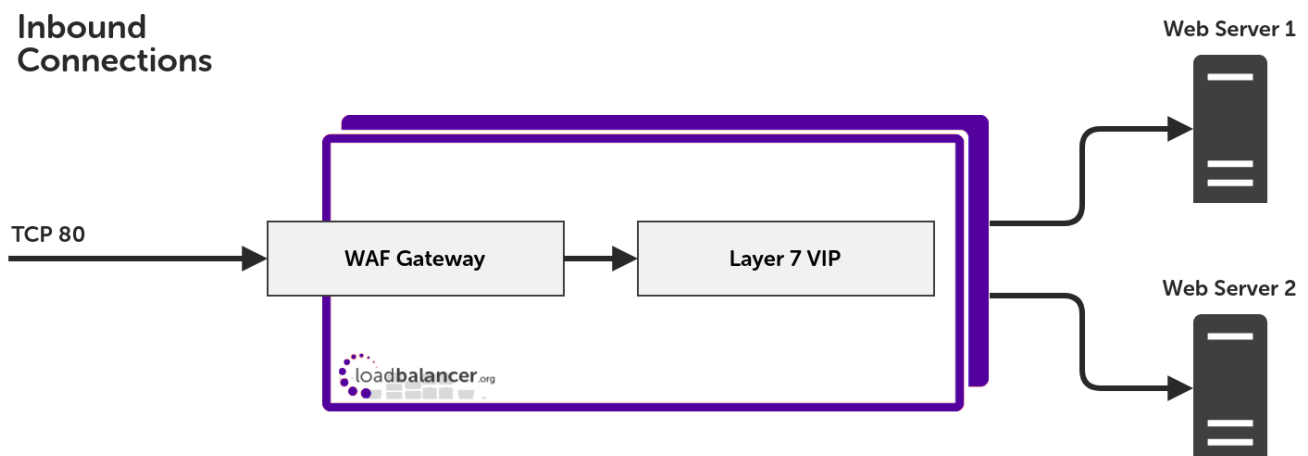
Implementation Concept

The load balancer supports the ability to configure multiple WAF gateways, one for each web application to be protected. Each WAF gateway is associated with a layer 7 VIP when created. On creation, the data path is automatically modified so that the WAF becomes the initial connection point for inbound client connections, as illustrated below:

Data flow before WAF is deployed:



Modified data flow once WAF is deployed:



Notes

- When configuring a WAF gateway on the load balancer, the associated layer 7 VIP must be selected from a dropdown list. This enables the WAF to be automatically configured to listen on the same TCP socket as the original layer 7 VIP. The WAF gateway is then automatically configured to forward packets to the original layer 7 VIP.
- Each WAF gateway is associated with one layer 7 VIP.
- Once the WAF gateway is configured, the **Label**, **IP Address**, **Port** and **Protocol** of the associated layer 7 VIP cannot be edited to ensure the association remains intact. If changes to these settings are required, take a backup copy of the WAF gateway's manual configuration (if one exists), remove the WAF, make the changes, and then recreate the WAF.

- Each WAF gateway is actually comprised of two component parts: an additional layer 7 VIP, which acts as the frontend to the WAF, and an Apache instance, of which ModSecurity is a module. Both are automatically created when the WAF gateway is configured.

Creating a New WAF Gateway

For reasons mentioned in the previous section, the layer 7 VIP must be created first (if it doesn't already exist), followed by the WAF gateway.

Step 1 - Create the Layer 7 VIP

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Virtual Services* and click **Add a new Virtual Service**.

Layer 7 - Add a new Virtual Service

Virtual Service		[Advanced +]
Label	<input type="text" value="Web-Cluster"/>	?
IP Address	<input type="text" value="192.168.110.46"/>	?
Ports	<input type="text" value="80"/>	?
Protocol		
Layer 7 Protocol	<input type="text" value="HTTP Mode"/>	?
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

2. Enter a suitable Label (name) for the VIP, e.g. **Web-Cluster**.
3. Enter a valid IP address, e.g. **192.168.110.46**.
4. Enter a valid port, e.g. **80**.
5. Click **Update**.

Step 2 - Define the Associated Real Servers (RIPs)

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Real Servers* and click **Add a new Real Server** next to the VIP just created.

Layer 7 Add a new Real Server - Web-Cluster

Label	<input type="text" value="Web1"/>	?
Real Server IP Address	<input type="text" value="192.168.110.241"/>	?
Real Server Port	<input type="text" value="80"/>	?
Re-Encrypt to Backend	<input type="checkbox"/>	?
Enable Redirect	<input type="checkbox"/>	?
Weight	<input type="text" value="100"/>	?

CancelUpdate

2. Enter a suitable Label (name) for the RIP, e.g. **Web1**.
3. Enter a valid IP address, e.g. **192.168.110.241**.
4. Enter a valid port, e.g. **80**.
5. Click **Update**.

Step 3 - Define the WAF Gateway

1. Using the WebUI, navigate to: *Cluster Configuration > WAF - Gateway* and click **Add a new WAF gateway**.

WAF - Add A New Gateway

Select Layer 7 Virtual Service	<input type="text" value="Web-Cluster"/>	?
WAF Label	<input type="text" value="WAF-Web-Cluster"/>	?
Ruleset	<input type="text" value="Core Rule Set 3.3.2"/>	?

CancelUpdate

2. Select the VIP created in step 1 in the dropdown.
3. The WAF label (name) field will be populated automatically, this can be changed if required.
4. The **Rule Set** will automatically choose the latest available stable version of the Core Rule Set. This should not ordinarily require changing.
5. Click **Update**.

Step 4 - Reload Services to Apply the New Settings

1. Click **System Overview** in the WebUI.
2. Reload the services (WAF and HAProxy) as prompted in the "Commit changes" message box.



Step 5 - View Configured Services



1. The original layer 7 VIP and the auto created layer 7 WAF frontend VIP are now displayed in the system overview as shown below:

System Overview ?

2023-02-01 16:53:53 UTC

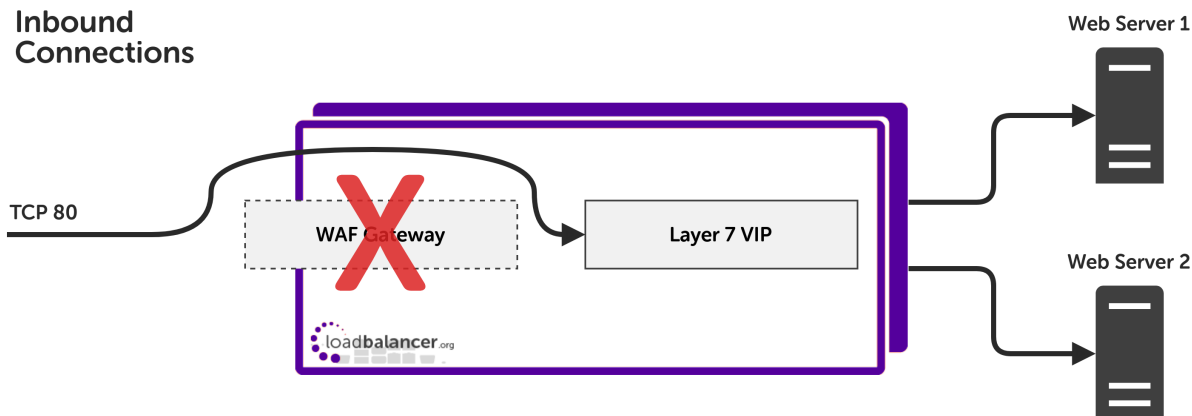
	VIRTUAL SERVICE	IP	PORTS	CONNS	PROTOCOL	METHOD	MODE	
↑	Web-Cluster	192.168.110.46	65435	0	HTTP	Layer 7	Proxy	
↑	WAF-Web-Cluster	192.168.110.46	80	0	HTTP	Layer 7	Proxy	

Fail Open and Fail Closed Designs

In the event that a WAF gateway should become unavailable for any reason, for example being overwhelmed by a denial of service attack, there are two response strategies that can be employed:

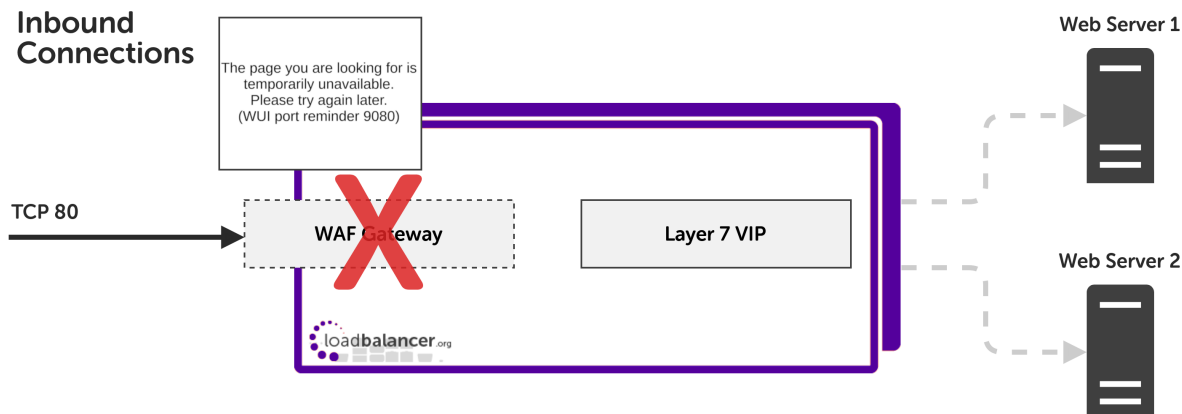
- **Fail open:** Traffic flow is allowed to continue by bypassing the failed WAF gateway, and service availability is maintained (*default*)

Inbound Connections



- **Fail closed:** Traffic flow is stopped until the WAF gateway can recover

Inbound Connections



Loadbalancer.org's top priority is ensuring that services are highly available, first and foremost. As such, the load balancer's default policy is to **fail open** in the event of a WAF gateway failure.

There may be scenarios where security is the **primary** concern, with high availability being a secondary

consideration in comparison. For example, if working with highly sensitive web traffic, it may be preferential to experience a service outage rather than allowing the service to be accessed insecurely by bypassing a WAF gateway.

Configuring a WAF Gateway for Fail Open Operation (Default)

Note

This is the default starting configuration for all WAF gateways.

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Virtual Services*.
2. Find the relevant pair of layer 7 virtual services: the original layer 7 VIP and the automatically created WAF frontend VIP, which is titled "WAF-..." by default.

Layer 7 - Virtual Services

Search..				Add a new Virtual Service	
Service Name	IP	Port	Config Type		
Web_Service	192.168.85.150	Ports 65435	Auto	Modify	
WAF-Web_Service	192.168.85.150	Ports 80	Auto	Modify	

3. Click **Modify** next to the "WAF-..." frontend VIP.
4. Scroll down to the *Fallback Server* section.
5. Set the *IP Address* to the address that the original layer 7 VIP is listening on, e.g. **192.168.85.150**.
6. Set the *Port* to the port that the original layer 7 VIP is listening on, e.g. **65435**.
7. Click **Update**.

Fallback Server			[Advanced +]
Disable Fallback	<input type="checkbox"/>		?
IP Address	<input type="text" value="192.168.85.150"/>		?
Port	<input type="text" value="65435"/>		?

8. Click *System Overview* in the WebUI.
9. Reload the services (WAF and HAProxy) as prompted in the "Commit changes" message box.

Configuring a WAF Gateway for Fail Closed Operation

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Virtual Services*.
2. Find the relevant pair of layer 7 virtual services: the original layer 7 VIP and the automatically created WAF frontend VIP, which is titled "WAF-..." by default.



Layer 7 - Virtual Services

<input type="text" value="Search.."/>				Add a new Virtual Service	
Service Name	IP	Port	Config Type		
Web_Service	192.168.85.150	Ports 65435	Auto	Modify	<input type="text"/>
WAF-Web_Service	192.168.85.150	Ports 80	Auto	Modify	<input type="text"/>

- Click **Modify** next to the "WAF-..." frontend VIP.
- Scroll down to the **Fallback Server** section.
- Set the **IP Address** to the local address of the fallback server, **127.0.0.1**.
- Set the **Port** to the port that the fallback server is listening on, **9081**.
- Click **Update**.

Fallback Server		[Advanced +]
Disable Fallback	<input type="checkbox"/>	?
IP Address	<input type="text" value="127.0.0.1"/>	?
Port	<input type="text" value="9081"/>	?

- Click **System Overview** in the WebUI.
- Reload the services (WAF and HAProxy) as prompted in the "Commit changes" message box.



Tip

It's possible to use a dedicated, external fallback server instead of the local instance on 127.0.0.1. It's also possible to customise the HTML of the built-in fallback page which is presented to users. For more information, refer to the [Fallback Server](#) section.

WAF Gateway Settings

Each WAF gateway has a variety of settings which can be configured through the WebUI. To access these settings, navigate to **Cluster Configuration > WAF - Gateway** and click **Modify** next to the WAF gateway in question.

Important

After modifying any WAF gateway settings, be sure to save the new configuration to disk by pressing the **Update** button and then apply the new configuration by reloading services as prompted in the "Commit changes" message box.

Disable Web Application Firewall

Default value: False.

When checked, this option completely disables all ModSecurity and CRS configuration for a given WAF gateway



(this includes disabling any manual WAF configuration, if present, while leaving the manual configuration itself intact). The proxy sandwich remains in place, allowing traffic to continue to flow in the same way, except without any WAF functionality present.

Ruleset

Default value: Core Rule Set 3.3.2.

The WAF rule set to use for a given WAF gateway can be selected from the dropdown list. There are currently two rule sets to choose from:

- **Core Rule Set 3.3.2** (default): the latest stable release of the OWASP ModSecurity Core Rule Set.
- **Core Rule Set 2**: a legacy option provided for backward compatibility with older WAF installations. Provides CRS version 2.2.9.

Paranoia Level

Default value: Paranoia Level 1.

Not available when the Core Rule Set 2 (legacy) is in use.

This can be set to paranoia level 1, 2, 3, or 4. Paranoia level 1 offers a baseline level of security with a minimal, or zero, need to tune away false positives. At the other end of the scale, paranoia level 4 offers the strongest level of security and features many additional rules, but is extremely likely to cause a large number of false positives, requiring a significant investment of time to tune them away.

See the section on [Paranoia Levels](#) for full details about this key concept.

Rule Engine Traffic Blocking

Default value: False.

Allows the WAF rule engine to take disruptive actions and block malicious looking traffic for this WAF gateway.

By default, the rule engine of a newly created WAF gateway is not able to block traffic. This is sometimes referred to as **detection only mode**. This means that WAF rule logic is processed in order to examine traffic but disruptive actions, e.g. "deny" and "drop", are **never executed**. As such, traffic is never blocked, even if it triggers rules and appears to be malicious.

One approach to configuring and tuning a WAF deployment is to leave the WAF gateway in detection only mode, pass known good traffic through the WAF (e.g. traffic from user testing), and then use the resulting log data to tune the WAF. This "tuning" is accomplished by writing rule exclusions to cover all false positives caused by the known good traffic. Once confident that all false positives have been accounted for, traffic blocking could then be enabled for the WAF gateway's rule engine. This takes the WAF gateway out of detection only mode and allows it to start actively blocking malicious looking traffic.

Process Request Data

Default value: True.



Instructs the WAF engine to buffer and process request bodies. This allows the data in request bodies to be inspected, for example the parameters of a POST request.

Process Response Data

Default value: False.

Instructs the WAF engine to buffer and process response bodies. This allows the data in response bodies to be inspected, for example an HTML response.

By default, a WAF gateway only processes request data, i.e. the data in requests coming in from clients. It's also possible to process response data, i.e. the data passed back **to** clients **from** the back end web application. This can be useful, for example, to catch instances of data leakage, such as an unintended SQL database error being passed back to the client (which may expose information about the type, version, and configuration of database software in use).

Inbound Anomaly Score

Default value: 20.

Sets the inbound anomaly score threshold: the cumulative anomaly score at which an inbound request will be blocked.

See the section on [Anomaly Scoring](#) for full details about this key concept.

Outbound Anomaly Score

Default value: 4.

Sets the outbound anomaly score threshold: the cumulative anomaly score at which an outbound response will be blocked.

See the section on [Anomaly Scoring](#) for full details about this key concept.

Audit Mode

Default value: False.

Enables the audit logging engine for all transactions passing through a WAF gateway.

Audit logs record full transaction data, **including full request bodies**. This information can be invaluable for troubleshooting particularly difficult issues.

Warning

Audit logs can grow **extremely large** very quickly. As such, it is **strongly recommended not to enable audit logging on a production machine**: doing so is likely to fill the logging disk partition and is **likely to cause disruptive issues on production machines**.

WAF Proxy Timeout

Default value: 120.

The Apache proxy service that hosts a WAF gateway has a 60 second timeout by default. This can be changed if required.

Enable Cache Acceleration

Default value: False.

While not directly related to WAF functionality, the proxy sandwich that hosts the WAF functionality also features a simple object cache. It will only cache objects that are HTML and below 64k in size, independent of any **cache** or **no-cache** options that your real servers may provide.

Enabling cache acceleration exposes the following cache-specific options:

- Force 'no-cache' override
- Location to exclude from the cache
- Cache object size

Double Login Enable

Default value: False.

Web Gateway Authentication

While not directly related to WAF functionality, the proxy sandwich that hosts the WAF functionality also features a simple web gateway / "double login" page. It supports the following authentication methods:

- Locally configured static user
- Google OpenID

Once enabled, users will be prompted for credentials when accessing the WAF:

A screenshot of a web login form titled "SECURE GATEWAY". The form has a light gray background with rounded corners. It contains two input fields: "Username" and "Password", both with light gray placeholder text. Below the "Password" field is a red "LOGIN" button with white text. The form is set against a dark gray background.

Secured by Loadbalancer.org

Enabling double login exposes the following double login-specific options:

- Location to protect
- Double login mode (Static user; OpenID Connect - Google)

- Static username
- Static password

WAF - Advanced Configuration

Note

These settings should not typically require changing.

The global WAF service has two advanced settings which can be configured through the WebUI. To access these settings, navigate to *Cluster Configuration > WAF - Advanced Configuration*.

The two advanced settings are *match limits*, which help avoid potential [regular expression denial of service](#) attacks by preventing PCRE (the underlying pattern matching library that evaluates regular expressions in ModSecurity / the WAF) from consuming huge amounts of system resources. PCRE uses a function called `match()` which it calls repeatedly, sometimes recursively. The match limits are imposed on the number of times this function is called during a match. The default values are both 250000, which is the minimum recommended value. A modern system (i.e. 4+ CPU cores, 8+ GB of RAM) could run without issue in production with match limits of 500000.

Important

After modifying any advanced WAF settings, be sure to save the new configuration to disk by pressing the **Set PCRE Match Limits** button and then apply the new configuration by reloading services as prompted in the "Commit changes" message box.

PCRE Match Limit

Default value: 250000.

Defines the global value of the `SecPcreMatchLimit` directive, which sets a maximum limit on the number of calls to the underlying match function when evaluating a regular expression in the WAF engine.

PCRE Match Limit Recursion

Default value: 250000.

Defines the global value of the `SecPcreMatchLimitRecursion` directive, which sets a maximum limit on the number of *recursive* calls to the underlying match function when evaluating a regular expression in the WAF engine.

WAF - Advanced Configuration

PCRE Match Limit

250000



PCRE Match Limit Recursion

250000



Set PCRE Match Limits

Working With the Core Rule Set

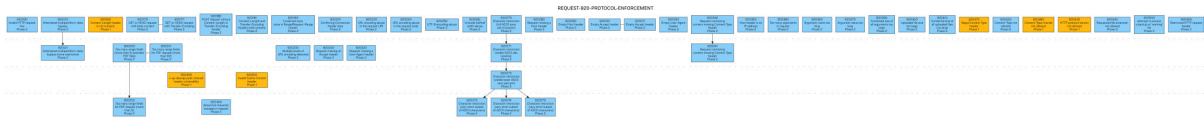


What is the Core Rule Set?

The OWASP® (Open Worldwide Application Security Project) [CRS \(Core Rule Set\)](#) is an open source collection of rules that work with ModSecurity® and compatible web application firewalls (WAFs). These rules are designed to provide easy to use, generic attack detection capabilities, with a minimum of false positives (false alerts), to your web application as part of a well balanced defense-in-depth solution.

Core Rule Set Map

A map of the Core Rule Set can be [downloaded as a PDF file](#) from our website. It shows each rule along with its ID number, a summary explanation, its phase, its paranoia level, and any relationships it has with other rules.



Anomaly Scoring

The Core Rule Set 3 is designed as an anomaly scoring rule set. This section explains what anomaly scoring is and how to use it.

Overview of Anomaly Scoring

Anomaly scoring, also known as "collaborative detection", is a scoring mechanism used in the Core Rule Set. It assigns a numeric score to HTTP transactions (requests and responses), representing how "anomalous" they appear to be. Anomaly scores can then be used to make blocking decisions. The default CRS blocking policy, for example, is to block any transaction that meets or exceeds a defined anomaly score threshold.

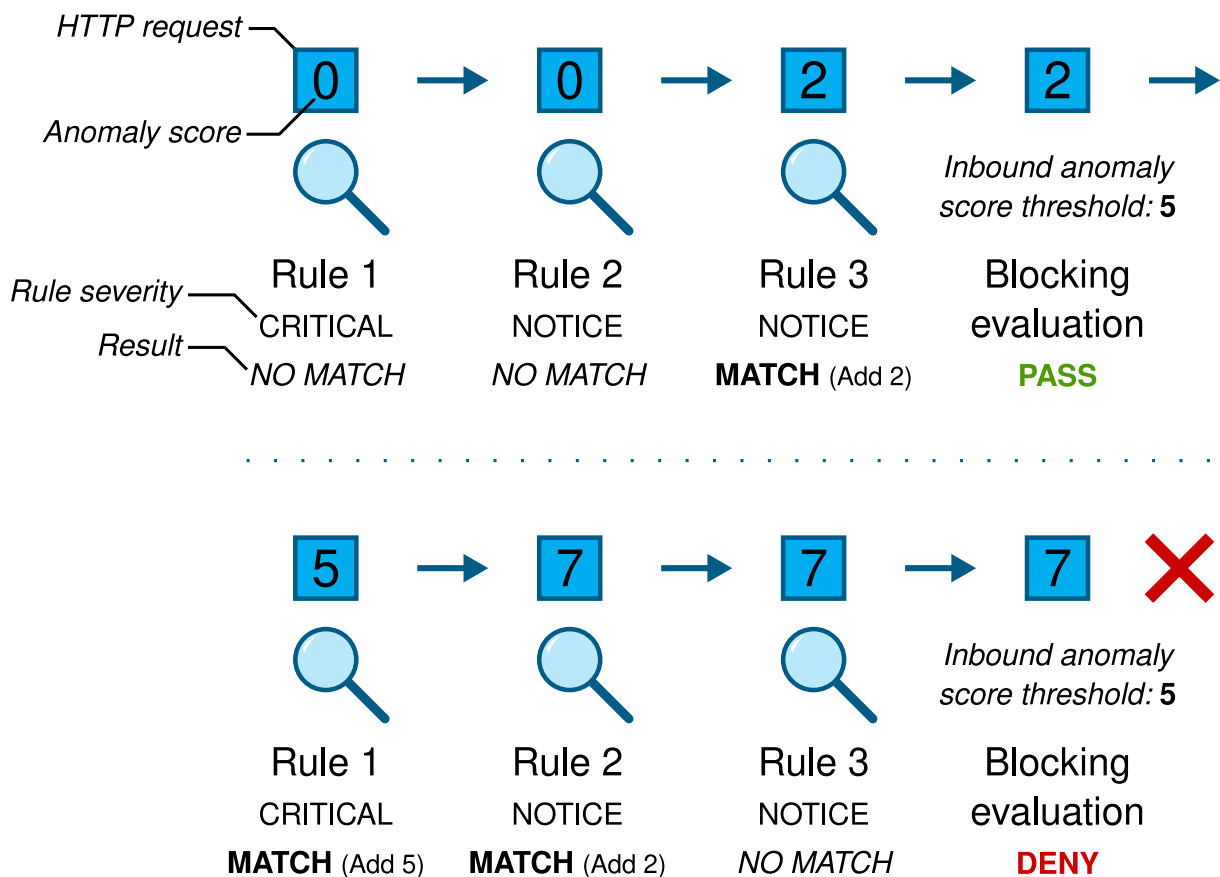
How Anomaly Scoring Mode Works

Anomaly scoring mode combines the concepts of *collaborative detection* and *delayed blocking*. The key idea to understand is that **the inspection/detection rule logic is decoupled from the blocking functionality**.

Individual rules designed to detect specific types of attacks and malicious behavior are executed. If a rule matches, no immediate disruptive action is taken (e.g. the transaction is not blocked). Instead, the matched rule contributes to a transactional **anomaly score**, which acts as a running total. The rules just handle detection, adding to the anomaly score if they match. In addition, an individual matched rule will typically log a record of the match for later reference, including the ID of the matched rule, the data that caused the match, and the URI that was being requested.

Once all of the rules that inspect **request** data have been executed, **blocking evaluation** takes place. If the anomaly score is greater than or equal to the inbound anomaly score threshold then the transaction is **denied**.

Transactions that are not denied continue on their journey.



Continuing on, once all of the rules that inspect *response* data have been executed, a second round of blocking evaluation takes place. If the *outbound* anomaly score is greater than or equal to the outbound anomaly score threshold then the transaction is *denied*.

Note

Having separate inbound and outbound anomaly scores and thresholds allows for request data and response data to be inspected and scored independently.

Summary of Anomaly Scoring Mode

To summarize, anomaly scoring mode in the CRS works like so:

1. Execute all *request* rules
2. Make a blocking decision using the *inbound* anomaly score threshold
3. Execute all *response* rules
4. Make a blocking decision using the *outbound* anomaly score threshold

Anomaly Score Thresholds


An anomaly score threshold is the cumulative anomaly score at which an inbound request or an outbound response will be blocked.

Most detected inbound threats carry an anomaly score of 5 (by default), while smaller violations, e.g. protocol and standards violations, carry lower scores. An anomaly score threshold of 7, for example, would require multiple rule matches in order to trigger a block (e.g. one "critical" rule scoring 5 plus a lesser-scoring rule, in order to reach

the threshold of 7). An anomaly score threshold of 10 would require at least two "critical" rules to match, or a combination of many lesser-scoring rules. **Increasing the anomaly score thresholds makes the CRS less sensitive** and hence less likely to block transactions.

Rule coverage should be taken into account when setting anomaly score thresholds. Different CRS rule categories feature different numbers of rules. SQL injection, for example, is covered by more than 50 rules. As a result, a real world SQLi attack can easily gain an anomaly score of 15, 20, or even more. On the other hand, a rare protocol attack might only be covered by a single, specific rule. If such an attack only causes the one specific rule to match then it will only gain an anomaly score of 5. If the inbound anomaly score threshold is set to anything higher than 5 then attacks like the one described will not be stopped. As such, a CRS installation should aim for an inbound anomaly score threshold of 5.

 **Warning** Increasing the anomaly score thresholds may allow some attacks to bypass the CRS rules.

 **Note** An outbound anomaly score threshold of 4 (the default) will block a transaction if any single response rule matches.

CRS uses two anomaly score thresholds, which can be defined for each WAF gateway. This is done using the WebUI, by navigating to: **Cluster Configuration > WAF - Gateway** and clicking **Modify** next to the relevant WAF. The two score thresholds are:

- Inbound Anomaly Score threshold
- Outbound Anomaly Score threshold

Severity Levels

Each CRS rule has an associated **severity level**. Different severity levels have different anomaly scores associated with them. This means that different rules can increment the anomaly score by different amounts if the rules match.

The four severity levels and their **default** anomaly scores are:

Severity Level	Anomaly Score
CRITICAL	5
ERROR	4
WARNING	3
NOTICE	2

For example, by default, a single matching **CRITICAL** rule would increase the anomaly score by 5, while a single matching **WARNING** rule would increase the anomaly score by 3.

Paranoia Levels

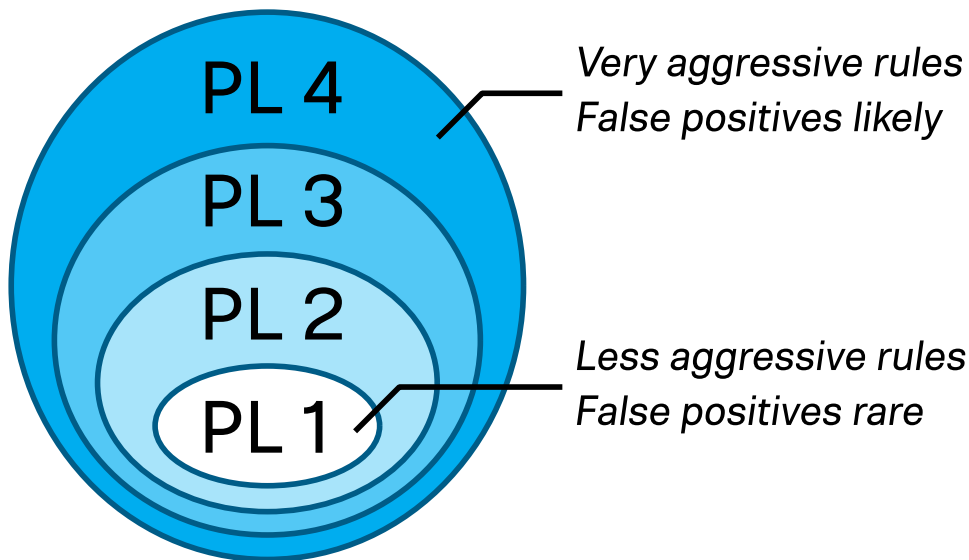
Paranoia levels are an essential concept when working with the Core Rule Set. This section explains the concept behind paranoia levels and how to work with them on a practical level.



Introduction to Paranoia Levels

The paranoia level (PL) makes it possible to define how aggressive the Core Rule Set is. Paranoia level 1 (PL 1) provides a set of rules that hardly ever trigger a false alarm (ideally never, but it can happen, depending on the local setup). PL 2 provides additional rules that detect more attacks (these rules operate *in addition* to the PL 1 rules), but there's a chance that the additional rules will also trigger new false alarms over perfectly legitimate HTTP requests.

This continues at PL 3, where more rules are added, namely for certain specialized attacks. This leads to even more false alarms. Then at PL 4, the rules are so aggressive that they detect almost every possible attack, yet they also flag a lot of legitimate traffic as malicious.



A higher paranoia level makes it harder for an attacker to go undetected. Yet this comes at the cost of more false positives: more false alarms. That's the downside to running a rule set that detects almost everything: your business / service / web application is also disrupted.

When false positives occur they need to be tuned away. In ModSecurity parlance: rule exclusions need to be written. A rule exclusion is a rule that disables another rule, either disabled completely or disabled partially only for certain parameters or for certain URIs. This means **the rule set remains intact** yet the CRS installation is no longer affected by the false positives.

Note

Depending on the complexity of the service (web application) in question and on the paranoia level, the process of writing rule exclusions can be a **substantial** amount of work.

For more information on this topic, see the section on [False Positives and Tuning](#).

Description of the Four Paranoia Levels

The CRS project views the four paranoia levels as follows:

Paranoia Level	Description
1	Baseline security with a minimal need to tune away false positives. This is CRS for everybody running an HTTP server on the internet. Please report any false positives encountered with a PL 1 system to support@loadbalancer.org .
2	Rules that are adequate when real user data is involved. Perhaps an off-the-shelf online shop. Expect to encounter false positives and learn how to tune them away.
3	Online banking level security with lots of false positives. From the CRS project's perspective, false positives are accepted and expected here, so it's important to learn how to write rule exclusions.
4	Rules that are so strong (or paranoid) they're adequate to protect the "crown jewels". To be used at one's own risk: be prepared to face a large number of false positives.

Choosing an Appropriate Paranoia Level

It's important to think about a service's security requirements. The difference between protecting a personal website and the admin gateway controlling access to an enterprise's Active Directory are very different. The paranoia level needs to be chosen accordingly, while also considering the resources (time) required to tune away false positives at higher paranoia levels.

Running at the highest paranoia level, PL 4, may seem appealing from a security standpoint, but *it could take many weeks to tune away the false positives encountered*. It is crucial to have enough time to fully deal with all false positives.

Warning

Failure to properly tune an installation runs the risk of exposing users to a vast number of false positives. This can lead to a poor user experience, and might ultimately lead to a decision to completely disable the WAF / Core Rule Set. As such, **setting a high PL in blocking mode without adequate tuning to deal with false positives is very risky**.

For an enterprise environment, consider developing an internal policy to map the risk levels and security needs of different assets to the minimum acceptable paranoia level to be used for them, for example:

- **Risk Class 0:** No personal data involved → PL 1
- **Risk Class 1:** Personal data involved, e.g. names and addresses → PL 2
- **Risk Class 2:** Sensitive data involved, e.g. financial/banking data; highest risk class → PL 3

Setting the Paranoia Level

To set the paranoia level for a WAF gateway, navigate to *Cluster Configuration > WAF - Gateway* and click **Modify** next to the WAF gateway in question.

From the **Paranoia Level** dropdown list, select the desired paranoia level.

Paranoia Level

Paranoia Level 1 ▼

Important

After modifying any WAF gateway settings, be sure to save the new configuration to disk by pressing the **Update** button and then apply the new configuration by reloading services as prompted in the "Commit changes" message box.

How Paranoia Levels Relate to Anomaly Scoring

It's important to understand that paranoia levels and CRS anomaly scoring (the CRS anomaly threshold/limit) are **two entirely different things with no direct connection**. The paranoia level controls the number of rules that are enabled while the anomaly threshold defines how many rules can be triggered before a request is blocked.

At the conceptual level, these two ideas *could* be mixed if the goal was to create a particularly granular security concept. For example, saying "we define the anomaly threshold to be 10, but we compensate for this by running at paranoia level 3, which we acknowledge brings more rule alerts and higher anomaly scores."

This is *technically* correct but it overlooks the fact that there are attack categories where CRS scores very low. For example, there is a plan to introduce a new rule to detect POP3 and IMAP injections: this will be a single rule, so, under normal circumstances, an IMAP injection would never score more than 5. Therefore, an installation running at an anomaly threshold of 10 could never block an IMAP injection, even if running at PL 3. In light of this, it's generally advised to **keep things simple and separate**: a CRS installation should aim for an anomaly threshold of 5 and a paranoia level as deemed appropriate.

False Positives and Tuning

When a **genuine** transaction causes a rule from the Core Rule Set to match in error it is described as a **false positive**. False positives need to be tuned away by writing **rule exclusions**, as this section explains.

What are False Positives?

The Core Rule Set provides **generic** attack detection capabilities. A fresh CRS deployment has no awareness of the web services that may be running behind it, or the quirks of how those services work. It is possible that **genuine** transactions may cause some CRS rules to match in error, if the transactions happen to match one of the generic attack behaviors or patterns that are being detected. Such a match is referred to as a **false positive**, or false alarm.

False positives are particularly likely to happen when operating at higher **paranoia levels**. While paranoia level 1 is designed to cause few, ideally zero, false positives, higher paranoia levels are increasingly likely to cause false positives. Each successive paranoia level introduces additional rules, with **higher** paranoia levels adding **more aggressive** rules. As such, the higher the paranoia level is the more likely it is that false positives will occur. That is the cost of the higher security provided by higher paranoia levels: the additional time it takes to tune away the increasing number of false positives.

Example False Positive

Imagine deploying the CRS in front of a WordPress instance. The WordPress engine features the ability to add HTML to blog posts (as well as JavaScript, if you're an administrator). Internally, WordPress has rules controlling which HTML tags are allowed to be used. This list of allowed tags has been studied heavily by the security community and it's considered to be a secure mechanism.

Consider the CRS inspecting a request with a URL like the following:



```
www.example.com/?wp_post=<h1>Welcome+To+My+Blog</h1>
```

At paranoia level 2, the `wp_post` query string parameter would trigger a match against an XSS attack rule due to the presence of HTML tags. CRS is unaware that the problem is properly mitigated on the server side and, as a result, the request causes a false positive and may be blocked. The false positive may generate an error log line like the following:

```
[Wed Jan 01 00:00:00.123456 2022] [:error] [pid 2357:tid 140543564093184] [client 10.0.0.1:0]
[client 10.0.0.1] ModSecurity: Warning. Pattern match
"<(?:a|abbr|acronym|address|applet|area|audioscope|b|base|basefont|bdo|bgsound|big|blackface|blink
|blockquote|body|bq|br|button|caption|center|cite|code|col|colgroup|comment|dd|del|dfn|dir|div|dl|dt|d
t|em|embed|fieldset|fn|font|form|frame|frameset|h1|head ..." at ARGS:wp_post. [file
"/etc/crs/rules/REQUEST-941-APPLICATION-ATTACK-XSS.conf"] [line "783"] [id "941320"] [msg "Possible
XSS Attack Detected - HTML Tag Handler"] [data "Matched Data: <h1> found within ARGS:wp_post:
<h1>welcome to my blog</h1>"] [severity "CRITICAL"] [ver "OWASP CRS/3.3.2"] [tag "application-
multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-xss"] [tag "OWASP CRS"] [tag
"capec/1000/152/242/63"] [tag "PCI/6.5.1"] [tag "paranoia-level/2"] [hostname "www.example.com"]
[uri "/" ] [unique_id "Yad-7q03dV56xYsnGhYJlQAAAAA"]
```

This example log entry provides lots of information about the rule match. Some of the key pieces of information are:

- The message from ModSecurity, which explains what happened and where:

```
ModSecurity: Warning. Pattern match "<(?:a|abbr|acronym ..." at ARGS:wp_post.
```

- The rule ID of the matched rule:

```
[id "941320"]
```

- The additional matching data from the rule, which explains precisely what caused the rule match:

```
[data "Matched Data: <h1> found within ARGS:wp_post: <h1>welcome to my
blog</h1>"]
```

Why are False Positives a Problem?

Alert Fatigue

If a system is prone to reporting false positives then the alerts it raises may be ignored. This may lead to real attacks being overlooked. For this reason, leaving false positives mixed in with real attacks is dangerous: the false positives should be resolved.

Sensitive Information and Regulatory Compliance

A false positive alert may contain sensitive information, for example usernames, passwords, and payment card data. Imagine a situation where a web application user has set their password to `/bin/bash`: without proper tuning, this input would cause a false positive every time the user logged in, writing the user's password to the error log file in plaintext as part of the alert.



It's also important to consider issues surrounding regulatory compliance. Data protection and privacy laws, like GDPR and CCPA, place strict duties and limitations on what information can be gathered and how that information is processed and stored. The unnecessary logging data generated by false positives can cause problems in this regard.

Poor User Experience

When working in strict blocking mode, false positives can cause legitimate user transactions to be blocked, leading to poor user experience. This can create pressure to disable the CRS or even to remove the WAF solution entirely, which is an unnecessary sacrifice of security for usability. The correct solution to this problem is to tune away the false positives so that they don't reoccur in the future.

Tuning Away False Positives

Directly Modifying CRS Rules

 **Warning** Making direct modifications to CRS rule files is a bad idea and is strongly discouraged.

It may seem logical to prevent false positives by modifying the offending CRS rules. If a detection pattern in a CRS rule is causing matches with genuine transactions then the pattern could be modified. **This is a bad idea.**

Directly modifying CRS rules essentially creates a fork of the rule set. Any modifications made would be undone by a rule set update, meaning that any changes would need to be continually reapplied by hand. This is a tedious, time consuming, and error-prone solution.

There are alternative ways to deal with false positives, as described below. These methods sometimes require slightly more effort and knowledge but they do not cause problems when performing rule set updates.

Rule Exclusions

Overview

The ModSecurity WAF engine has flexible ways to tune away false positives. It provides several **rule exclusion** (RE) mechanisms which allow rules to be modified **without** directly changing the rules themselves. This makes it possible to work with third-party rule sets, like the Core Rule Set, by adapting rules as needed while leaving the rule set files intact and unmodified. This allows for easy rule set updates.

Two fundamentally different types of rule exclusions are supported:

- **Configure-time rule exclusions:** Rule exclusions that are applied once, at **configure-time** (e.g. when (re)starting or reloading ModSecurity, or the server process that holds it). For example: "remove rule X at startup and never execute it."

This type of rule exclusion takes the form of a ModSecurity directive, e.g. [SecRuleRemoveById](#).

- **Runtime rule exclusions:** Rule exclusions that are applied at **runtime** on a per-transaction basis (e.g. exclusions that can be conditionally applied to some transactions but not others). For example: "if a transaction is a POST request to the location 'login.php', remove rule X."

This type of rule exclusion takes the form of a [SecRule](#).

Note

Runtime rule exclusions, while granular and flexible, have a computational overhead, albeit a small one. A runtime rule exclusion is an extra SecRule which must be evaluated for every transaction.

In addition to the two **types** of exclusions, rules can be excluded in two different **ways**:

- **Exclude the entire rule/tag:** An entire rule, or entire category of rules (by specifying a tag), is removed and will not be executed by the rule engine.
- **Exclude a specific variable from the rule/tag:** A *specific variable* will be excluded from a specific rule, or excluded from a category of rules (by specifying a tag).

These two methods can also operate on multiple individual rules, or even entire rule categories (identified either by tag or by using a range of rule IDs).

The combinations of rule exclusion types and methods allow for writing rule exclusions of varying granularity. Very coarse rule exclusions can be written, for example "remove all SQL injection rules" using [SecRuleRemoveByTag](#). Extremely granular rule exclusions can also be written, for example "for transactions to the location 'web_app_2/function.php', exclude the query string parameter 'user_id' from rule 920280" using a SecRule and the action `ctl:ruleRemoveTargetById`.

The different rule exclusion types and methods are summarized in the table below, which presents the main ModSecurity directives and actions that can be used for each type and method of rule exclusion:

	Exclude entire rule/tag	Exclude specific variable from rule/tag
Configure-time	SecRuleRemoveById* SecRuleRemoveByTag	SecRuleUpdateTargetById SecRuleUpdateTargetByTag
Runtime	ctl:ruleRemoveById** ctl:ruleRemoveByTag	ctl:ruleRemoveTargetById ctl:ruleRemoveTargetByTag

*Can also exclude ranges of rules or multiple space separated rules.

**Can also exclude ranges of rules.

Important

It is not currently possible to exclude **phase 1** rules using **runtime** rule exclusions on Loadbalancer.org appliances. Configure-time rule exclusions must be used instead. This is a known limitation of the WAF implementation. The [Core Rule Set Map](#) can be used to lookup the phase of a CRS rule. More information on rule phases can be found in the [ModSecurity Reference Manual](#).

Note

There's also a third group of rule exclusion directives and actions, the use of which is discouraged. As well as excluding rules "ById" and "ByTag", it's also possible to exclude "ByMsg" ([SecRuleRemoveByMsg](#), [SecRuleUpdateTargetByMsg](#), [ctl:ruleRemoveByMsg](#), and [ctl:ruleRemoveTargetByMsg](#)). This excludes rules based on the message they write to the

error log. These messages can be dynamic and may contain special characters. As such, trying to exclude rules by message is difficult and error-prone.

Rule Tags

CRS rules typically feature multiple tags, grouping them into different categories. For example, a rule might be tagged by attack type ('attack-rce', 'attack-xss', etc.), by language ('language-java', 'language-php', etc.), and by platform ('platform-apache', 'platform-unix', etc.).

Tags can be used to remove or modify entire categories of rules all at once, but some tags are more useful than others in this regard. Tags for **specific** attack types, languages, and platforms may be useful for writing rule exclusions. For example, if lots of the SQL injection rules are causing false positives but SQL isn't in use anywhere in the back end web application then it may be worthwhile to remove all CRS rules tagged with "attack-sqli" (`SecRuleRemoveByTag attack-sqli`).

Some rule tags are **not** useful for rule exclusion purposes. For example, there are generic tags like "language-multi" and "platform-multi": these contain hundreds of rules across the entire CRS, and they don't represent a meaningful rule property to be useful in rule exclusions. There are also tags that categorize rules based on well known security standards, like CAPEC and PCI DSS (e.g. 'capec/1000/153/267', 'PCI/6.5.4'). These tags may be useful for informational and reporting purposes but are not useful in the context of writing rule exclusions.

Excluding rules using tags may be more useful than excluding using rule ranges in situations where a category of rules is spread across multiple files. For example, the "language-php" rules are spread across several different rule files (both inbound and outbound rule files).

Rule Ranges

As well as rules being tagged using different categories, CRS rules are organized into files by general category. In addition, CRS rule IDs follow a consistent numbering convention. This makes it easy to remove unwanted types of rules by removing ranges of rule IDs. For example, the file `REQUEST-913-SCANNER-DETECTION.conf` contains rules related to detecting well known scanners and crawlers, which all have rule IDs in the range 913000-913999. All of the rules in this file can be easily removed using a configure-time rule exclusion, like so:

```
SecRuleRemoveById "913000-913999"
```

Excluding rules using rule ranges may be more useful than excluding using tags in situations where tags are less relevant or where tags vary across the rules in question. For example, a rule range may be the most appropriate solution if the goal is to remove all rules contained in a single file, regardless of how the rules are tagged.

Support for Regular Expressions

Most of the configure-time rule exclusion directives feature some level of support for using regular expressions. This makes it possible, for example, to exclude a dynamically named variable from a rule. The directives with support for regular expressions are:

- `SecRuleRemoveByTag`

A regular expression is used for the tag match. For example, `SecRuleRemoveByTag "injection"` would match both "attack-injection-generic" and "attack-injection-php".



- `SecRuleRemoveByMsg`

A regular expression is used for the message match. For example, `SecRuleRemoveByMsg "File Access"` would match both "OS File Access Attempt" and "Restricted File Access Attempt".

- `SecRuleUpdateTargetById`, `SecRuleUpdateTargetByTag`, `SecRuleUpdateTargetByMsg`

A regular expression can optionally be used in the target specification by enclosing the regular expression in forward slashes. This is useful for dealing with dynamically named variables, like so:

```
SecRuleUpdateTargetById 942440 "!REQUEST_COOKIES:/^uid_.*/"
```

This example would exclude request cookies named "uid_0123456", "uid_6543210", etc. from rule 942440.

 **Note**

The "ctl" action for writing runtime rule exclusions does **not** support any use of regular expressions. This is a known limitation of the ModSecurity rule engine.

Example 1 (`SecRuleRemoveById`)

(Configure-time RE. Exclude entire rule.)

Scenario: Rule 933151, "PHP Injection Attack: Medium-Risk PHP Function Name Found", is causing false positives. The web application behind the WAF makes no use of PHP. As such, it is deemed safe to tune away this false positive by completely removing rule 933151.

Rule Exclusion:

```
# CRS Rule Exclusion: 933151 - PHP Injection Attack: Medium-Risk PHP Function Name Found
SecRuleRemoveById 933151
```

Example 2 (`SecRuleRemoveByTag`)

(Configure-time RE. Exclude entire tag.)

Scenario: Several different parts of a web application are causing false positives with various SQL injection rules. None of the web services behind the WAF make use of SQL, so it is deemed safe to tune away these false positives by removing all the SQLi detection rules.

Rule Exclusion:

```
# CRS Rule Exclusion: Remove all SQLi detection rules
SecRuleRemoveByTag attack-sqli
```

Example 3 (`SecRuleUpdateTargetById`)

(Configure-time RE. Exclude specific variable from rule.)

Scenario: The content of a POST body parameter named "wp_post" is causing false positives with rule 941320,

"Possible XSS Attack Detected - HTML Tag Handler". Removing this rule entirely is deemed to be unacceptable: the rule is not causing any other issues, and the protection it provides should be retained for everything apart from "wp_post". It is decided to tune away this false positive by excluding "wp_post" from rule 941320.

Rule Exclusion:

```
# CRS Rule Exclusion: 941320 - Possible XSS Attack Detected - HTML Tag Handler
SecRuleUpdateTargetById 941320 "!ARGS:wp_post"
```

Example 4 (*SecRuleUpdateTargetByTag*)

(Configure-time RE. Exclude specific variable from rule.)

Scenario: The values of request cookies with random names of the form "uid_<STRING>" are causing false positives with various SQL injection rules. It is decided that it is not a risk to allow SQL-like content in cookie values, however it is deemed unacceptable to disable the SQLi detection rules for anything apart from the request cookies in question. It is decided to tune away these false positives by excluding only the problematic request cookies from the SQLi detection rules. A regular expression is to be used to handle the random string portion of the cookie names.

Rule Exclusion:

```
# CRS Rule Exclusion: Exclude the request cookies "uid_<STRING>" from the SQLi detection rules
SecRuleUpdateTargetByTag attack-sqli "!REQUEST_COOKIES:/^uid_.*/"
```

Example 5 (*ctl:ruleRemoveById*)

(Runtime RE. Exclude entire rule.)

Scenario: Rule 920230, "Multiple URL Encoding Detected", is causing false positives at the specific location "/webapp/function.php". This is being caused by a known quirk in how the web application has been written, and it cannot be fixed in the application. It is deemed safe to tune away this false positive by removing rule 920230 for that specific location only.

Rule Exclusion:

```
# CRS Rule Exclusion: 920230 - Multiple URL Encoding Detected
SecRule REQUEST_URI "@beginsWith /webapp/function.php" \
    id:1000, \
    phase:1, \
    pass, \
    nolog, \
    ctl:ruleRemoveById=920230"
```

Example 6 (*ctl:ruleRemoveByTag*)

(Runtime RE. Exclude entire tag.)

Scenario: Several different locations under "/web_app_1/content" are causing false positives with various SQL injection rules. Nothing under that location makes any use of SQL, so it is deemed safe to remove all the SQLi

detection rules for that location. Other locations **may** make use of SQL, however, so the SQLi detection rules **must** remain in place everywhere else. It has been decided to tune away the false positives by removing all the SQLi detection rules for locations under "/web_app_1/content" only.

Rule Exclusion:

```
# CRS Rule Exclusion: Remove all SQLi detection rules
SecRule REQUEST_URI "@beginsWith /web_app_1/content" \
    "id:1010,\
    phase:1,\
    pass,\
    nolog,\
    ctl:ruleRemoveByTag=attack-sqli"
```

Example 7 (*ctl:ruleRemoveTargetById*)

(Runtime RE. Exclude specific variable from rule.)

Scenario: The content of a POST body parameter named "text_input" is causing false positives with rule 941150, "XSS Filter - Category 5: Disallowed HTML Attributes", at the specific location "/dynamic/new_post". Removing this rule entirely is deemed to be unacceptable: the rule is not causing any other issues, and the protection it provides should be retained for everything apart from "text_input" at the specific problematic location. It is decided to tune away this false positive by excluding "text_input" from rule 941150 for location "/dynamic/new_post" only.

Rule Exclusion:

```
# CRS Rule Exclusion: 941150 - XSS Filter - Category 5: Disallowed HTML Attributes
SecRule REQUEST_URI "@beginsWith /dynamic/new_post" \
    "id:1020,\
    phase:1,\
    pass,\
    nolog,\
    ctl:ruleRemoveTargetById=941150;ARGS:text_input"
```

Example 8 (*ctl:ruleRemoveTargetByTag*)

(Runtime RE. Exclude specific variable from rule.)

Scenario: The values of request cookie "uid" are causing false positives with various SQL injection rules when trying to log in to a web service at location "/webapp/login.html". It is decided that it is not a risk to allow SQL-like content in this specific cookie's values for the login page, however it is deemed unacceptable to disable the SQLi detection rules for anything apart from the specific request cookie in question at the login page only. It is decided to tune away these false positives by excluding only the problematic request cookie from the SQLi detection rules, and only when accessing "/webapp/login.html".

Rule Exclusion:

```
# CRS Rule Exclusion: Exclude the request cookie "uid" from the SQLi detection rules
SecRule REQUEST_URI "@beginsWith /webapp/login.html" \
    "id:1030,\
```



```
phase:1, \
pass, \
nolog, \
ctl:ruleRemoveTargetByTag=attack-sqli;REQUEST_COOKIES:uid"
```



Tip

It's possible to write a conditional rule exclusion that tests something other than just the request URI. Conditions can be built which test, for example, the source IP address, HTTP request method, HTTP headers, and even the day of the week.

Multiple conditions can also be chained together to create a logical AND by using ModSecurity's [chain](#) action. This allows for creating powerful rule logic like "for transactions that are from source IP address 10.0.0.1 AND that are for location '/login.html', exclude the query string parameter 'user_id' from rule 920280". Extremely granular and specific rule exclusions can be written, in this way.

Adding Custom WAF Configuration

Rule exclusions and custom rules can be added to a WAF gateway. To access this functionality:

1. Using the WebUI, navigate to: **Cluster Configuration > WAF - Manual Configuration**.
2. From the dropdown list, select the name of the WAF gateway in question.

By default, a WAF gateway's manual configuration contains a set of commented-out examples. The examples can be uncommented and adapted for use. Alternatively, all of the example text can safely be deleted to give a clean slate.

To add rule exclusions or custom rules, write or paste the custom content directly into the text box.

WAF - Manual Configuration

WAF-Web_Service ▾

```
1  # CRS Rule Exclusion: 932150 - Remote Command Execution: Direct Unix Command
2  #                               Execution
3  #
4  # The argument "text_input" contains arbitrary user input. Natural text like
5  # "ping pong is great" causes a false positive with rule 932150. Exclude
6  # "text_input" from the rule.
7  SecRuleUpdateTargetById 932150 "!ARGS:text_input"
8
9
10
11 # All traffic hitting the WAF comes from the public internet, so deny all access
12 # to locations under "/app/admin_portal".
13 SecRule REQUEST_URI "@beginsWith /app/admin_portal/" \
14     "id:1000,\
15     phase:1,\
16     deny,\
17     log,\
18     msg:'Deny admin portal access through the WAF, i.e. for external traffic'"
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
```

Update

❗ Important

After modifying any WAF manual configuration, be sure to save the new configuration to disk by pressing the **Update** button and then apply the new configuration by reloading services as prompted in the "Commit changes" message box.

Per-Transaction Resource Limits

WAFs can be susceptible to simple denial-of-service attacks. Bombarding a WAF with certain types of HTTP requests can cause serious problems:

- Requests with large bodies (excluding file uploads) consume a significant amount of CPU time while the WAF rules are executed against the body data.
- Requests with large bodies consume a large amount of memory while they're buffered for inspection.
- Requests with very large headers consume a significant amount of CPU time while the WAF rules are executed against the header data.
- Large file uploads consume a large amount of disk space while they're buffered during a transaction.

A WAF can thus be attacked from multiple angles: attempting to exhaust its CPU, memory, and disk space. As such, it is imperative to control and limit the resources that a **single HTTP transaction** can consume. This hardens the WAF against these kinds of attacks.

Default Limits



The following default resource limits are enforced for all WAF gateways:

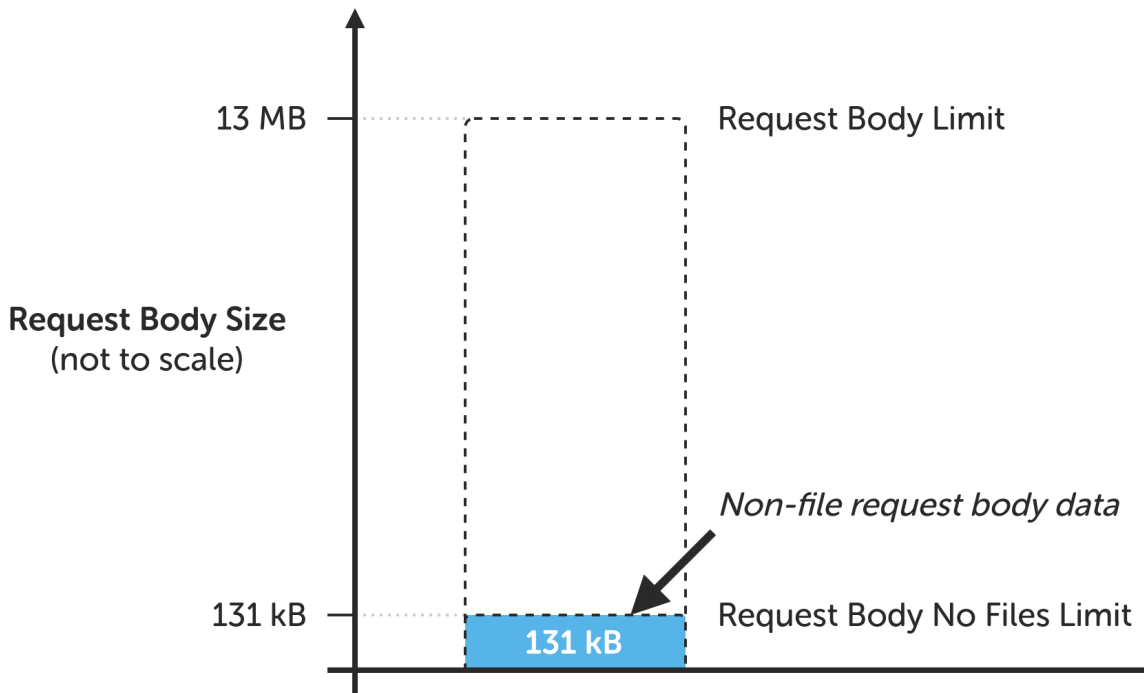
- Maximum request body size (including file uploads): **13 MB** (13107200 bytes)
- Maximum request body size (excluding file uploads): **131 kB** (131072 bytes)
- Maximum amount of request body to store in memory before switching to disk: **131 kB** (131072 bytes)
- Maximum size of a single HTTP request header field: **8 kB** (8190 bytes)

 **Tip**

The default limits for the maximum non-file request body size and the maximum amount of memory to use to buffer a request body are equal. This means that non-file request bodies under the size limit will always be buffered in memory, which is fast. Only request bodies of file uploads exceeding 131 kB will be streamed to disk, which is much slower.

Example 1

As an example, under the default limits, it is permissible to submit an HTTP request with up to 131 kB of non-file upload data, e.g. data from filling in form fields:

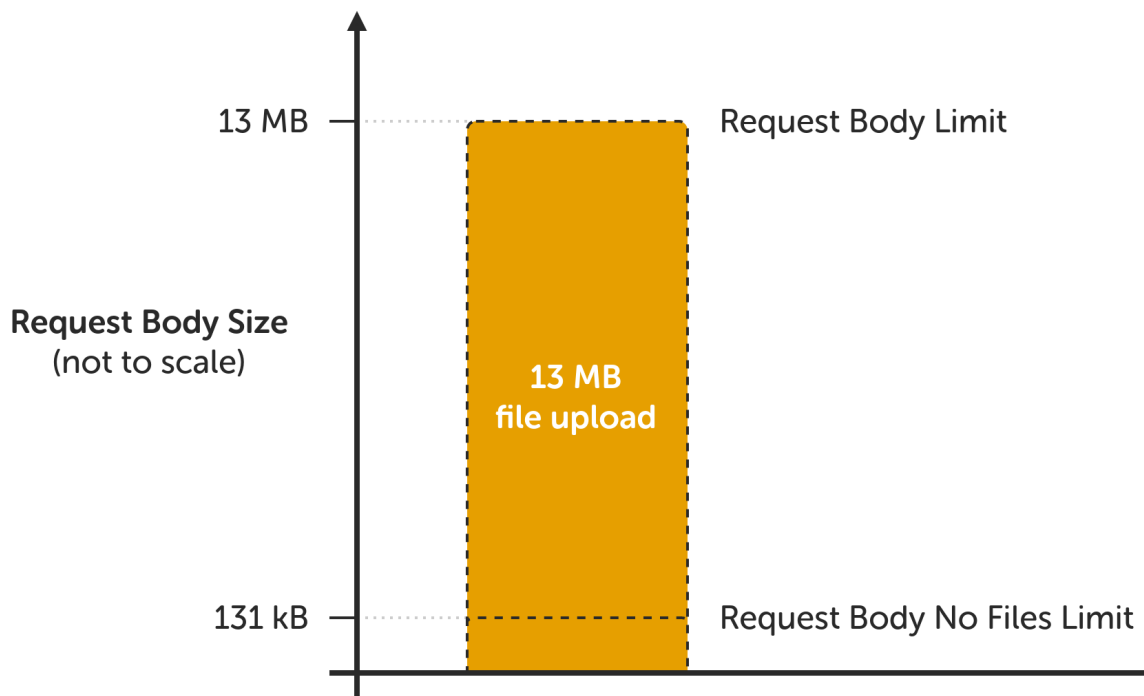


A request that exceeds the limit will:

- receive a **400 Bad Request** response status code if the WAF engine is **enabled**
- pass through the WAF gateway without issue if the WAF engine is **not enabled**, i.e. is in **detection only mode**

Example 2

As another example, under the default limits, it is permissible to submit an HTTP request with up to 13 MB of file upload data, either one large file or multiple smaller files:



A request that exceeds the limit will:

- receive a **500 Internal Server Error** response status code if the WAF engine is **enabled**
- pass through the WAF gateway without issue if the WAF engine is **not enabled**, i.e. is in **detection only mode**

Handling Large HTTP Requests

When working with some web applications, it can be necessary to handle HTTP requests that exceed the default size limits in one or more ways.

Large File Uploads

The maximum request body size, **including** file uploads, should be as large as the largest file upload that the web application should accept. If the web application should accept file uploads that are **larger** than the default 13 MB limit then the limit should be increased.

As an example, to increase the limit to **20 MB** for a given WAF gateway, add the following directive to the WAF gateway's manual configuration:

```
# Buffer request bodies of up to 20 MB in size.  
# Allows for requests to contain up to 20 MB of file upload data.  
SecRequestBodyLimit 20000000
```

For additional granularity, a **Location** directive can be used to increase the limit for certain locations only. For example:

```
# For the admin portal upload page only, buffer request bodies of up to 500 MB  
# in size. Allows for requests to contain up to 500 MB of file upload data.  
<Location "/admin-portal/upload.php">
```

```
SecRequestBodyLimit 500000000
</Location>
```

Note

Request body handling has a hard upper limit of 1 GB.

Requests Containing Large Amounts of Non-File Upload Data

The maximum request body size, **excluding** file uploads, should be as small as is practical. An example of non-file upload data would be data from filling in form fields. Unlike file upload data, **non**-file upload data is inspected: the WAF rules are executed against the data. For large chunks of data, rule execution can use a significant amount of CPU time.

If a web application should accept request bodies containing **more** than the default limit of 131 kB of non-file upload data then the limit should be increased. Some web applications transfer state data or small files in this way, which can often exceed the 131 kB limit where this technique is used.

Warning

Increasing the non-file upload request body size limit makes the WAF more susceptible to denial-of-service attacks. This limit should **only** be increased if necessary and should still be kept as small as possible.

As an example, to increase the limit to **200 kB** for a given WAF gateway, add the following directive to the WAF gateway's manual configuration:

```
# Buffer request bodies, excluding file uploads, of up to 200 kB in size.
# Allows for requests to contain up to 200 kB of non-file upload data.
SecRequestBodyNoFilesLimit 200000
```

Because increasing this limit makes a WAF more susceptible to denial-of-service attacks, it is a very good idea to increase the limit only where it is needed. For additional granularity, a **Location** directive can be used to increase the limit for certain locations only. For example:

```
# For the user profile avatar upload functionality only, buffer request bodies
# containing up to 200 kB of non-file upload data. Allows users to upload
# avatars up to 200 kB in size.
<Location "/user/profile/avatar-upload.php">
    SecRequestBodyNoFilesLimit 200000
</Location>
```

Note

The limit for the maximum amount of memory to use to buffer a request body can be set using the `SecRequestBodyInMemoryLimit` directive. The default values of the `SecRequestBodyNoFilesLimit` and `SecRequestBodyInMemoryLimit` directives are equal.

Requests with Massive Header Fields

If a web application requires the use of massive HTTP request header fields, with any single header field exceeding the default size limit of 8 kB, then the limit should be increased.



As an example, to increase the limit to **13 kB** for a given WAF gateway, which should be large enough to let through many known large authentication headers which are seen very occasionally, add the following directive to the WAF gateway's manual configuration:

```
# Allow HTTP request header fields of up to 13 kB in size.  
# Should be large enough to let through many known large authentication headers.  
LimitRequestFieldSize 13000
```

Note that the `LimitRequestFieldSize` directive cannot be defined inside a `Location` directive, so being granular with this setting is not possible.

Note

To handle HTTP request header fields above **15 kB** in size it's necessary to increase HAProxy's **Request buffer length**. This can be found under *Cluster Configuration > Layer 7 - Advanced Configuration*. This is required as HAProxy becomes the limiting factor, not Apache.

WAF Gateway Error Logs

Logging Mechanism Overview

If a transaction does not cause any ModSecurity (WAF) rules to match then nothing is written to the WAF error log.

If a transaction **does** cause rules to match:

- A record of each rule match is written to the WAF error log.
- If a transaction's inbound or outbound anomaly score is high enough to reach either the inbound or outbound anomaly score thresholds then a record of this is written to the WAF error log, stating the action taken, if any (e.g. "Access denied with code 403").
- A summary of the transaction's anomaly score is written to the WAF error log, stating the makeup of the anomaly score total per-paranoia level and per-rule category.

Tip

If a rule is **expected** to match and should **not** log those matches for some reason (e.g. it's a helper rule), the `no log` action can be used to prevent a rule from creating error log output and cluttering up the log.

Viewing the Error Logs

To view the error logs for a given WAF gateway:

1. Using the WebUI, navigate to: **Logs > WAF Error**.
2. From the dropdown list, select the name of the WAF gateway in question.

Default View

After selecting a WAF gateway from the dropdown list, the default view displays error log lines in their raw, unedited format. The most recent 500 log lines are displayed in **reverse chronological order**, with the most recent log entry shown at the top.

Error WAF-Web_Service ▾

Simple

Breakdown

Fixes

Empty Log

Download

```
[Wed Feb 01 16:35:47.658635 2023] [:error] [pid 6905:tid 140248700925696] [client 192.168.85.1:0]
[client 192.168.85.1] ModSecurity: Warning. Operator GE matched 5 at TX:inbound_anomaly_score. [file
"/etc/modsecurity.d/rules_3-3-2/RESPONSE-980-CORRELATION.conf"] [line "91"] [id "980130"] [msg
"Inbound Anomaly Score Exceeded (Total Inbound Score: 5 -
SQLI=0,XSS=0,RFI=0,LFI=0,RCE=5,PHPI=0,HTTP=0,SESS=0): individual paranoia level scores: 5, 0, 0, 0"]
[ver "OWASP CRS/3.3.2"] [tag "event-correlation"] [hostname "example.com"] [uri "/login"] [unique_id
"Y9qU4zW4XsSqM8XUvX2LCQAAAAQ"]
[Wed Feb 01 16:35:47.657121 2023] [:error] [pid 6905:tid 140248700925696] [client 192.168.85.1:0]
[client 192.168.85.1] ModSecurity: Access denied with code 403 (phase 2). Operator GE matched 5 at
TX:anomaly_score. [file "/etc/modsecurity.d/rules_3-3-2/REQUEST-949-BLOCKING-EVALUATION.conf"] [line
"100"] [id "949110"] [msg "Inbound Anomaly Score Exceeded (Total Score: 5)"] [severity "CRITICAL"]
[ver "OWASP CRS/3.3.2"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag
"attack-generic"] [hostname "example.com"] [uri "/login"] [unique_id "Y9qU4zW4XsSqM8XUvX2LCQAAAAQ"]
```

Simple View

When the **Simple** button is pressed, the most recent 1000 error log lines are shown in chronological order (matching the log file itself), with the oldest log entry shown at the top.

Log entries are broken up across multiple lines to aid readability. Individual log entries are separated by horizontal lines.

Error WAF-Web_Service ▾

Simple

Breakdown

Fixes

Empty Log

Download

```
[Wed Feb 01 16:35:47.654387 2023]
[:error]
[pid 6905:tid 140248700925696]
[client 192.168.85.1:0]
[client 192.168.85.1]
• ModSecurity: Warning. Matched phrase "bin/bash" at ARGS:usr.
[file "/etc/modsecurity.d/rules_3-3-2/REQUEST-932-APPLICATION-ATTACK-RCE.conf"]
[line "500"]
• [id "932160"]
[msg "Remote Command Execution: Unix Shell Code Found"]
[data "Matched Data: bin/bash found within ARGS:usr: /bin/bash"]
[severity "CRITICAL"]
[ver "OWASP CRS/3.3.2"]
[tag "application-multi"]
[tag "language-shell"]
[tag "platform-unix"]
[tag "attack-rce"]
[tag "paranoia-level/1"]
[tag "OWASP CRS"]
[tag "capec/1000/152/248/88"]
[tag "PCI/6.5.2"]
[hostname "example.com"]
[uri "/login"]
[unique_id "Y9qU4zW4XsSqM8XUvX2LCQAAAAQ"]
```

Breakdown View

When the **Breakdown** button is pressed, a summary of which matched rules are present in the error log is presented. The information presented is:



- Occurrences: The number of times a given event is present in the log, e.g. "how many times rule X has matched for host name Y at location Z".
- Rule ID: The ID number of the rule that matched.
- Hostname: The host name that the matches were against. Useful to break up data if multiple services are sitting behind a single WAF gateway.
- Severity (optional): If present, the severity of the rule that matched (as described in the [Severity Levels](#) section).
- URI: The location that the match was against.

Error WAF-Web_Service ▾

Simple

Breakdown

Fixes

Empty Log

Download

Occurrences | Rule ID | Hostname | [-Rule severity (optional)-] | URI

1 942100 example.com [- (5) CRITICAL-] /login

1 932160 example.com [- (5) CRITICAL-] /login

Fixes View

When the **Fixes** button is pressed, a best efforts, automated list of rule exclusions are generated. These rule exclusions are based on the assumption that only known, good traffic has passed through (and hence been logged by) the WAF gateway. It is thus assumed that the error log contains only *false positives*, which the load balancer attempts to generate rule exclusions to resolve.

⚠ Caution The ***Fixes*** view is not a substitute for a proper tuning process. Writing meaningful rule exclusions requires an understanding of the web service behind the WAF gateway, which the script that generates the automated rule exclusions can never provide.

Error WAF-Web_Service ▾

Simple

Breakdown

Fixes

Empty Log

Download

<LocationMatch "(?i)^/login\$">
SecRuleRemoveById 932160
SecRuleRemoveById 942100
</LocationMatch>

Breakdown of a Log Entry

A single, full error log entry from the **Simple** view is presented below:

```
[Fri Dec 03 12:14:06.323315 2021]
    [:error]
    [pid 2547:tid 140172380346112]
    [client 192.168.85.1:0]
    [client 192.168.85.1]
    • ModSecurity: Warning. Pattern match
"(?:^|=)\\\\s*(?:{\\\\\\\\s*\\\\\\\\(\\\\\\\\s*\\\\\\\\w+=(?:[\\\\\\\\\\\\s]*\\\\\\\\$.*\\\\\\\\$.*|<.*>.*|\\\\\\\\'.*\\\\\\\\'|\\\\\\\\".*
\\\\\\\\")\\\\\\\\s+|!\\\\\\\\s*\\\\\\\\$)*\\\\\\\\s*(?:'|\\\\\\\\")*(?:[\\\\\\\\?\\\\\\\\*\\\\\\\\[\\\\\\\\]\\\\\\\\(\\\\\\\\)\\\\\\\\-
\\\\\\\\|+\\\\\\\\w'\\\\\\\\"\\\\\\\\.\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\]+)?[\\\\\\\\\\\\\\\\\\\\\\\\'\\\\\\\\"]*(?:l[\\\\\\\\\\\\\\\\\\\\\\\\'\\\\\\\\"]*(?:s(?:[\\\\\\\\\\\\\\\\\\\\\\\\'\\\\\\\\"]*(?:b[\\\\\\\\\\\\\\\\\\\\\\\\'\\\\\\\\"]*
_\\\\\\\\\\\\\\\\\\\\\\\\'\\\\\\\\"]*r[\\\\\\\\\\\\\\\\\\\\\\\\'\\\\\\\\"]*e[\\\\\\\\\\\\\\\\\\\\\\\\'\\\\\\\\"]*l[\\\\\\\\\\\\\\\\\\\\\\\\' ..." at ARGS:text_input.
```

```
[file "/etc/modsecurity.d/rules_3-3-2/REQUEST-932-APPLICATION-ATTACK-RCE.conf"]
[line "463"]
• [id "932150"]
  [msg "Remote Command Execution: Direct Unix Command Execution"]
  [data "Matched Data: ping found within ARGS:text_input: ping pong is great!"]
  [severity "CRITICAL"]
  [ver "OWASP_CRS/3.3.2"]
  [tag "application-multi"]
  [tag "language-shell"]
  [tag "platform-unix"]
  [tag "attack-rce"]
  [tag "paranoia-level/1"]
  [tag "OWASP_CRS"]
  [tag "capec/1000/152/248/88"]
  [tag "PCI/6.5.2"]
  [hostname "example.com"]
  [uri "/"]
  [unique_id "YaoKDrMWmz0GZzK9ZmMCAAAEA"]
```

The most important parts and their meanings are as follows:

- [Fri Dec 03 12:14:06.323315 2021]

The date and time that the log entry was written.

- [client 192.168.85.1]

The source IP address of the transaction.

- ModSecurity: Warning. Pattern match "(?:^|=)..." at ARGS:text_input.

A summary message from ModSecurity describing what has happened and where.

- [id "932150"]

The rule ID of the rule that matched.

- [msg "Remote Command Execution: Direct Unix Command Execution"]

A summary message from the matched rule describing what happened.

- [data "Matched Data: ping found within ARGS:text_input: ping pong is great!"]

Additional data from the rule describing what happened.

- [severity "CRITICAL"]

The severity of the rule that matched (as described in the [Severity Levels](#) section).



- `[ver "OWASP_CRS/3.3.2"]`

The Core Rule Set version of the rule that matched.

- `[tag "platform-unix"]`

A tag used to categorise the type of rule or type of attack it is designed to detect.

- `[tag "paranoia-level/1"]`

A tag used to report the paranoia level of the rule that matched.

- `[tag "OWASP_CRS"]`

A tag used to report that the matched rule was part of the Core Rule Set.

- `[hostname "example.com"]`

The host name used in the transaction.

- `[uri "/"]`

The location requested by the transaction.

- `[unique_id "YaoKDmrMWmz0GZzK9ZmMCAAAAEA"]`

The transaction's unique ID, which ties together all log entries relating to a single transaction.

An error log entry from the *Simple* view showing the outcome of inbound blocking evaluation is presented below:

```
[Fri Dec 03 12:14:06.327738 2021]
  [:error]
  [pid 2547:tid 140172380346112]
  [client 192.168.85.1:0]
  [client 192.168.85.1]
    • ModSecurity: Access denied with code 403 (phase 2). Operator GE matched 5 at
TX:anomaly_score.
  [file "/etc/modsecurity.d/rules_3-3-2/REQUEST-949-BLOCKING-EVALUATION.conf"]
  [line "93"]
    • [id "949110"]
      [msg "Inbound Anomaly Score Exceeded (Total Score: 5)"]
      [severity "CRITICAL"]
      [ver "OWASP_CRS/3.3.2"]
      [tag "application-multi"]
      [tag "language-multi"]
      [tag "platform-multi"]
      [tag "attack-generic"]
      [hostname "example.com"]
      [uri "/"]
      [unique_id "YaoKDmrMWmz0GZzK9ZmMCAAAAEA"]
```

The most important unique parts and their meanings are as follows:

- `ModSecurity: Access denied with code 403 (phase 2). Operator GE matched 5 at TX:anomaly_score.`

A summary message from ModSecurity describing what has happened (access denied with status code 403 Forbidden) and why (anomaly score reached threshold of 5).

- `[msg "Inbound Anomaly Score Exceeded (Total Score: 5)"]`

A summary message from the matched (blocking evaluation) rule describing what happened, in a slightly more readable way.

An error log entry from the *Simple* view showing an event correlation summary is presented below:

```
[Fri Dec 03 12:14:06.329017 2021]
  [:error]
  [pid 2547:tid 140172380346112]
  [client 192.168.85.1:0]
  [client 192.168.85.1]
  • ModSecurity: Warning. Operator GE matched 5 at TX:inbound_anomaly_score.
    [file "/etc/modsecurity.d/rules_3-3-2/RESPONSE-980-CORRELATION.conf"]
    [line "91"]
  • [id "980130"]
    [msg "Inbound Anomaly Score Exceeded (Total Inbound Score: 5 -
SQLI=0,XSS=0,RFI=0,LFI=0,RCE=5,PHPI=0,HTTP=0,SESS=0): individual paranoia level scores: 5, 0, 0,
0"]
    [ver "OWASP CRS/3.3.2"]
    [tag "event-correlation"]
    [hostname "example.com"]
    [uri "/"]
    [unique_id "YaoKDrMWmz0GZzK9ZmMCAAAEA"]
```

The most important unique part and its meaning is as follows:

- `[msg "Inbound Anomaly Score Exceeded (Total Inbound Score: 5 -
SQLI=0,XSS=0,RFI=0,LFI=0,RCE=5,PHPI=0,HTTP=0,SESS=0): individual paranoia
level scores: 5, 0, 0, 0"]`

A summary message from the matched (event correlation) rule giving a full breakdown of the transaction's anomaly score, both by paranoia level and category. The paranoia level breakdown is given in ascending paranoia level (i.e. PL 1, PL 2, PL 3, PL 4).

Chapter 8 - Real Server Health Monitoring & Control

Configuring Health Checks

The appliance supports a comprehensive range of Real Server health check methods and options. These range from simple ping checks to much more complex negotiate options to determine that the underlying daemon/service is running and responding correctly. The specific options available depend on whether services are deployed at Layer 4 or Layer 7, details of both are covered in the following sections.

Health Checks for Layer 4 Services

At layer 4, Real Server health checking is performed by Ldirectord. To configure health checks, use the WebUI menu option: **Cluster Configuration > Layer 4 - Virtual Services**, then click **Modify** next to the VIP to be configured. The health check options available depend on the check type selected.

Health Checks	
Check Type	<div>Connect to port ▾</div> <div>Negotiate</div> <div>Connect to port</div> <div>ping server</div> <div>External script</div> <div>No checks, always Off</div> <div>No checks, always On</div> <div>5 Connects, 1 Negotiate</div> <div>10 Connects, 1 Negotiate</div>
Check Port	
Feedback	
Feedback Method	
Fallback Server	


Note



For new Layer 4 VIPs the default check type is **Connect to Port**.

The following table details the health check types available and the associated options for each:

Health Check Type	Option	Description
Negotiate		Sends a request and looks for a specific response.
	Check Port	The port to monitor. If specified this setting overrides the default which is the Real Server port. For multiport VIPs where the Real Server port field is left blank, the default checkport is the first in the list. This can be changed using this field if required. Note that for DR mode, the port cannot be specified at the Real Server level, so the port specified for the VIP is used.

Health Check Type	Option	Description
	<i>Protocol</i>	<p>Set the protocol for the health check. The options are:</p> <ul style="list-style-type: none"> • HTTP - use HTTP as the negotiate protocol • HTTPS - use HTTPS as the negotiate protocol • HTTP Proxy - Use an HTTP proxy check • FTP - use FTP as the negotiate protocol • IMAP (IPv4 only) - use IMAP as the negotiate protocol • IMAPS (IPv4 only) - use IMAPS as the negotiate protocol • POP - use POP as the negotiate protocol • POPS - use POPS as the negotiate protocol • LDAP (IPv4 only) - use LDAP as the negotiate protocol • SMTP - use SMTP as the negotiate protocol • NNTP (IPv4 only) - use NNTP as the negotiate protocol • DNS - use DNS as the negotiate protocol • MySQL (IPv4 only) - use MySQL as the negotiate protocol (also requires username/password) • SIP - use SIP as the negotiate protocol • Simple TCP - Sends a request string to the server and checks the response • RADIUS (IPv4 only) - use RADIUS as the negotiate protocol
	<i>Virtual Host</i>	<p>Used when using a HTTP or HTTPS negotiate check. Sets the host header used in the HTTP request. In the case of HTTPS this generally needs to match the common name of the SSL certificate.</p>

Health Check Type	Option	Description
	<i>Request to Send</i>	<p>The use of this parameter varies with the protocol selected in the negotiate <i>Protocol</i>.</p> <ul style="list-style-type: none"> • With protocols such as HTTP and FTP, this should be the object to request from the server. Bare filenames will be requested from the web or FTP root. • With DNS, this should be either a name to look up in an A record, or an IP address to look up in a PTR record. • With databases, this should be a SQL SELECT query. The <i>Response Expected</i> field is not used by the SQL health check since the data returned is not read, the answer must simply be one or more rows. • With LDAP, this should be the search base for the query. The load balancer will perform an (ObjectClass=*) search relative to this base. • With Simple TCP, this should be a string to send verbatim to the server.
	<i>Response Expected</i>	<p>This is the response that must be received for the check to be a success. If the string matches anywhere in the response data, the negotiate check is considered a success. For a DNS check this should be any one of the A record's addresses or any one of the PTR record's names.</p> <div>  Note <p>For HTTP and HTTPS, if <i>Response Expected</i> is left blank, the appliance will check the location specified in <i>Request To Send</i> (if blank the root will be checked) and will consider all HTTP 2xx (usually HTTP 200) and HTTP 3xx response statuses as valid and the server will be marked as up. All other responses including no response or a timeout will be considered invalid and the server will be marked as down.</p> </div>
	<i>Login</i>	The username when authentication is required.
	<i>Password</i>	The password when authentication is required.
	<i>Database Name</i>	The database to use for the MySQL check.
	<i>Radius Secret</i>	The RADIUS secret string to use for the RADIUS check.
Connect to Port		Attempt to make a TCP connection to the specified port.
	<i>Check Port</i>	See above.
Ping Server		Sends an ICMP echo request packet to the Real Server.
External Script		Use a custom file (typically a script but can also be a binary) for the health check. Select the script from the <i>External script</i> dropdown. For more information on using custom external health check scripts, please refer to External Health Check Scripts .

Health Check Type	Option	Description
	<i>Check Port</i>	See above.
	<i>External Script</i>	<p>Select the required external check script from the dropdown. For more information on creating and using custom external health check scripts, please refer to External Health Check Scripts.</p> <div> <p> Note</p> <p>By default the Microsoft SQL external health check is not available in the dropdown. This health check requires several Microsoft related pre-requisites such as the Microsoft Linux ODBC driver, and these must first be installed and configured. To install the pre-requisites and configure the required settings, at the console or via an SSH session, login as "root" and run the following command:</p> <pre>\$ lb_mssql -i</pre> <p>Once completed, the additional option "ms-sql-check" will appear in the External Script dropdown.</p> </div> <div> <p> Note</p> <p>"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: Local Configuration > Security. You'll need to Set Appliance Security Mode to Custom, configure the required option(s) and click Update.</p> </div>
No Checks Always Off		No health checks, all Real Servers are assumed to be down.
No Checks Always On		No health checks, all Real Servers are assumed to be up.
5 Connects, 1 Negotiate		Do 5 connect checks followed by 1 negotiate check. The appropriate negotiate check must be selected and configured (see above).
10 Connects, 1 Negotiate		Do 10 connect checks followed by 1 negotiate check. The appropriate negotiate check must be selected and configured (see above).

Health Checks for Layer 7 Services

At layer 7, Real Server health checking is performed by HAProxy. To configure health checks, use the WebUI menu option: **Cluster Configuration > Layer 7 - Virtual Services**, then click **Modify** next to the VIP to be configured. The health check options available depend on the check type selected and whether the **[Advanced]** option is clicked. When clicked, the advanced options for each check type are displayed, when clicked again, they are hidden.

Health Checks

[Advanced +]

Health Checks

Connect to port

Negotiate HTTP (GET)
Negotiate HTTP (HEAD)
Negotiate HTTPS (GET)
Negotiate HTTPS (HEAD)
Negotiate HTTP (OPTIONS)
Negotiate HTTPS (OPTIONS)
Connect to port
External script
MySQL
No checks, always On

ACL Rules

Type	Bool	URL/Text

Add Rule



Note

For new Layer 7 VIPs the default check type is *Connect to Port*.

The following table details the health check types available and the associated options for each:

Health Check Type	Option	Description
Negotiate HTTP/HTTPS (GET)		Scan the file/page specified in <i>Request to Send</i> , and check the returned data for the <i>Response Expected</i> string.
	<i>Request to Send</i>	The file/page to open. This can also be the name of a server-side script that's used to check the health of the backend application.
	<i>Response Expected</i>	<p>The content expected in the specified file/page. The <i>Response Expected</i> can also be any valid regular expression statement. The result can be inverted by selecting "Not Equals".</p> <div> <div>Note</div> <p>For HTTP and HTTPS, if <i>Response Expected</i> is left blank, the appliance will check the location specified in Request To Send (if blank the root will be checked) and will consider all HTTP 2xx (usually HTTP 200) and HTTP 3xx response statuses as valid and the server will be marked as up. All other responses including no response or a timeout will be considered invalid and the server will be marked as down.</p> </div> <div> <div>Note</div> <p>It's possible to escape characters in the response expected. For example, if you wanted to look for "success" (including the quotes), specify \"success\" in <i>Response Expected</i>.</p> </div>
	Advanced > <i>Check Port</i>	The port to monitor. If specified this setting overrides the default which is the Real Server port. For multiport VIPs where the Real Server port field is left blank, the default checkport is the first in the list. This can be changed using this field if required.

Health Check Type	Option	Description
	Advanced > <i>Username</i>	Specify a username if authentication is required.
	Advanced > <i>Host Header</i>	If the Real Server is configured to require a Host header, specify the value here.
	Advanced > <i>Password</i>	Specify a password if authentication is required.
Negotiate HTTP/HTTPS (HEAD)		Request the page headers of the page specified in <i>Request to Send</i> .
	<i>Request to Send</i>	See above.
	Advanced > <i>Check Port</i>	See above.
	Advanced > <i>Username</i>	See above.
	Advanced > <i>Host Header</i>	See above.
	Advanced > <i>Password</i>	See above.
Negotiate HTTP/HTTPS (OPTIONS)		Request the options of the page specified in <i>Request to Send</i> .
	<i>Request to Send</i>	See above.
	Advanced > <i>Check Port</i>	See above.
	Advanced > <i>Username</i>	See above.
	Advanced > <i>Host Header</i>	See above.
	Advanced > <i>Password</i>	See above.
Connect to port		Attempt to make a TCP connection to the specified port.
	Advanced > <i>Check Port</i>	See above.
External Script		Use a custom file (typically a script but can also be a binary) for the health check. Select the script from the <i>Check Script</i> dropdown. For more information on using custom external health check scripts, please refer to External Health Check Scripts .

Health Check Type	Option	Description
	<i>Check Script</i>	<p>Select the required external check script from the dropdown. For more information on creating and using custom external health check scripts, please refer to External Health Check Scripts.</p> <div>  Note <p>By default the Microsoft SQL external health check is not available in the dropdown. This health check requires several Microsoft related per-requisites such as the Microsoft Linux ODBC driver, and these must first be installed and configured. To install the prerequisites and configure required settings, at the console or via an SSH session, login as "root" and run the following command:</p> <pre>\$ lb_mssql -i</pre> <p>Once completed, the additional option "ms-sql-check" will appear in the External Script dropdown.</p> </div> <div>  Note <p>"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: Local Configuration > Security. You'll need to Set Appliance Security Mode to Custom, configure the required option(s) and click Update.</p> </div>
MySQL		<p>The check consists of sending two MySQL packets, one Client Authentication packet, and one QUIT packet to correctly close the MySQL session. It then parses the MySQL Handshake Initialization packet and/or Error packet. It is a basic but useful test and does not produce error nor aborted connect on the server. However, it requires adding an authorization in the MySQL table, like this:</p> <pre>USE mysql; INSERT INTO user (Host,User) values ('',''); FLUSH PRIVILEGES;</pre>
	Advanced > Username	This is the user that will be used to connect to the database to verify it's running successfully.
No checks, always On		No health checks, all Real Servers are assumed to be up.

External Health Check Scripts

Custom health checks can be created and modified using the WebUI.

Default Scripts

By default, 5 external scripts are available:

- SMTP
- Ping_IPv4_or_IPv6
- POP3_or_IMAP
- Exchange
- TCP_half_open

These are available in the *External Script* dropdown when the *Check Type* for a VIP is set to **External Script** as shown below:

The screenshot shows the 'Health Checks' configuration page. The 'Health Checks' section has a dropdown menu set to 'External script'. A dropdown menu is open, showing the following options: SMTP, Ping_IPv4_or_IPv6 (highlighted), POP3_or_IMAP, Exchange, and TCP_half_open. Below this, the 'ACL Rules' section is visible, showing a table with columns: Type, Bool, and URL/Text. An 'Add Rule' button is located at the bottom right of the ACL Rules section.

Adding Additional Health Check Scripts

New scripts can be created either by using the script templates or by uploading files from an external source.

Using Script Templates

To create a new script from template:

1. Using the WebUI, navigate to *Cluster Configuration > Health Check Scripts* and click **Add New Health Check**.

The screenshot shows the 'Health Check Details' form. It has three fields: 'Name' with the value 'Multi-Port-Check', 'Type' with the value 'Virtual Service', and 'Template' with the value 'Multi-port-check.sh'. Each field has a help icon (question mark) to its right.

2. Specify an appropriate **Name** for the health check, e.g. **Multi-Port-Check**.
3. Set **Type** to **Virtual Service**.
4. Using the **Template** dropdown select an appropriate template from the **Virtual Service** section of the list, e.g. **Multi-port-check.sh**.
5. Modify the script if required.

6. Click **Update**.

Once the health check has been added, it will appear in the Health Check Scripts list below the 5 default scripts as shown below:

Health Check Scripts

			Add New Health Check
			Upload Existing Health Check
Health Check Name	Type	In-use	
SMTP	VIP	-	Modify Delete
Ping_IPv4_or_IPv6	VIP	-	Modify Delete
POP3_or_IMAP	VIP	-	Modify Delete
Exchange	VIP	-	Modify Delete
TCP_half_open	VIP	-	Modify Delete
Multi-Port-Check	VIP	-	Modify Delete

The new script will also appear in the *External Script* dropdown when the *Check Type* for a VIP is set to **External Script**:

Health Checks

[Advanced +]

Health Checks

External script

?

Check Script

Ping_IPv4_or_IPv6

?

ACL Rules

SMTP

Ping_IPv4_or_IPv6

POP3_or_IMAP

Exchange

TCP_half_open

Multi-Port-Check

Type

Bool

URL/Text

edirect

Add Rule

?

Uploading External Files

To create a new script by uploading an external file:

1. Using the WebUI, navigate to *Cluster Configuration > Health Check Scripts* and click **Upload Existing Health Check**.

Health check Details

Name:

?

Type:

☒ Virtual Service

?

☐ GSLB

Contents:

?

Secondary node contents:

?

File is binary:

☐

?

Cancel

Update

- Specify an appropriate **Name** for the health check, e.g. **Check-Control-Servers**.
- Set **Type** to **Virtual Service**.
- Click the **Choose File** button next to **Contents**.
- Browse to and select the required file, e.g. **control-server-check.sh**.

Note

If you have an HA Pair and the secondary node requires a different health check, click the **Choose File** button next to **Secondary Node Contents** and browse to and select the required file.

- If the file is binary, enable the **File is binary** checkbox - this will prevent the editor window being displayed.
- Click **Update**.

Once the health check has been added, it will appear in the Health Check Scripts list and in the **External Script** dropdown as explained in [Using Script Templates](#) above.

Note

For additional Layer 4 health check options such as **Check interval** and **Failure Count**, please refer to [Layer 4 - Advanced Configuration](#). For Layer 7, please refer to [Layer 7 - Advanced Configuration](#).

Testing External Health Check Scripts at the Command Line

Health check scripts use 4 passed parameters. These 4 values represent **Virtual Service IP Address**, **Virtual Service Port**, **Real Server IP Address** and **Real Server Port**. If a script does not use all 4 values, for example the ping.sh script, then a zero should be entered as a place-holder.

Note

The **Skeleton** and **Example** script templates provide additional information about the 4 passed parameters.

```
# ./<check-script-name> <$1> <$2> <$3> <$4>
```

Examples:



```
# ./SMTP-check.sh 192.168.1.1 25 192.168.1.10 25
```

```
# ./ping.sh 192.168.1.1 0 192.168.1.10 0
```

to check the return value, use the command:

```
# echo $?
```

A return value of 0 means the check has passed, any other value means it has failed.

Simulating Health Check Failures

It may not always be possible to take a server offline to check that health checks are working correctly. In these cases, firewall rules can be used. The following rules can be configured at the console, using SSH or via the WebUI option *Local Configuration > Execute a Shell Command*.

Note

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, configure the required option(s) and click **Update**.

To block access to a particular Real Server port:

```
iptables -A OUTPUT -p tcp --dport <Check Port> -d <REAL-SERVER-IP> -j DROP
```

e.g. `iptables -A OUTPUT -p tcp --dport 80 -d 192.168.65.60 -j DROP`

To re-enable access to a particular Real Server port:

```
iptables -D OUTPUT -p tcp --dport <Check Port> -d <REAL-SERVER-IP> -j DROP
```

e.g. `iptables -D OUTPUT -p tcp --dport 80 -d 192.168.65.60 -j DROP`

Note

Make sure that blocking rules are deleted after testing & verification is complete!

Disabling Health Checks

In some cases it may be desirable to completely disable health checking and simply assume that the Real Servers are up and working correctly. This can be configured by setting the health check option to **No Checks, Always On** - this applies to both layer 4 and layer 7 services.

Fallback Server



The fallback server is activated under the following conditions for both Layer 4 & Layer 7 Virtual Services:

- When all associated Real Servers have failed their health check
- When all associated Real Servers have been taken offline using the System Overview page in the WebUI

The fallback page can be provided in the following ways:

- Using the load balancer's built in NGINX local fallback page
- Using a separate server to host the fallback page
- Using a layer 7 VIP

Local Fallback Server

The appliance has a built in fallback server that uses NGINX. The local fallback page can be modified using the WebUI menu option: **Maintenance > Fallback Page**:

Fallback Page

```
1 <html>
2 <head>
3 <title>The page is temporarily unavailable</title>
4 <style>
5 body { font-family: Tahoma, Verdana, Arial, sans-serif; }
6 </style>
7 </head>
8 <body bgcolor="white" text="black">
9 <table width="100%" height="100%">
10 <tr>
11 <td align="center" valign="middle">
12 The page you are looking for is temporarily unavailable.<br/>
13 Please try again later.<br/>
14 (WUI port reminder 9080)
15 </td>
16 </tr>
17 </table>
18 </body>
19 </html>
20
```

Notes

1. The local fallback server is an NGINX instance that by default listens on all appliance IP's on port 9081. If you want to change the port, IP address or protocol that the fallback server listens on, please refer to [Service Socket Addresses](#).
2. You can use any valid HTML for the default page, simply copy and paste the required HTML into the Fallback Page.
3. If you are using the load balancer for your holding page and your Real Servers are all offline then the local NGINX server is exposed to hacking attempts, if you are concerned about this you can use a separate dedicated internal server for the fallback.

Using a Separate Dedicated Server

For DR mode the fallback server must be listening on the same port as the VIP (port re-mapping is not possible with DR mode). Also, the "ARP Problem" must be solved for the dedicated fallback server. For more information, please refer to [The ARP Problem](#).



For NAT mode the default gateway of the fallback server must be the load balancer. For an HA clustered pair, use a floating IP for the default gateway to allow it to move between Primary and Secondary.

Using a Layer 7 VIP

It's also possible to set the fallback server to be a layer 7 VIP. This is especially useful in WAN/DR site environments. It also enables an external fallback server to be easily configured for Layer 4 VIPs without having to comply with the requirements mentioned in the previous section. To do this, create a layer 7 fallback VIP and configure your fallback server as an associated RIP. Then enable the MASQ option for the Layer 4 VIP and set the fallback VIP as its fallback server. If all servers are down, requests will then be routed via the Layer 7 VIP to your fallback server. If the layer 4 VIP is multi-port, specify "0" as the port for the fallback server. Requests will then be forwarded to the correct port.

Configuring A Real Server as the Fallback Server

It's possible to configure one of the Real Servers as the fallback server. This can be useful for example when all servers are very busy and health checks start to fail simply because the response is taking longer than the configuration allows. In this case, traffic will still be sent to one of the Real Servers rather than to a separate fallback page.

Configuring Primary / Secondary Real Servers

If you want to setup a VIP that sends all traffic to a primary server and only sends traffic to a secondary server if the primary server fails, configure the VIP with the primary server as a RIP, and the secondary server as the fallback server.

Configuring Email Alerts for Virtual Services

Email alerts can be configured for layer 4 and layer 7 Virtual Services. This enables emails to be sent when one or more of the associated Real Servers fail their health check and also when they subsequently start to pass their health check.

Layer 4





For layer 4 Virtual Services, settings can be configured globally for all VIPs or individually per VIP.

Global Layer 4 Email Settings

Once configured, these settings apply to all layer 4 VIPs by default.

To configure global email alert settings for layer 4 services:

1. Using the WebUI, navigate to: **Cluster Configuration > Layer 4 Advanced Configuration**.

Email Alert Source Address	<input type="text" value="lb1@loadbalancer.org"/>	
Email Alert Destination Address	<input type="text" value="alerts@loadbalancer.org"/>	
Auto-NAT	<input type="text" value="off"/> ▾	
Multi-threaded	<input type="text" value="yes"/> ▾	

Update

2. Enter an appropriate email address in the *Email Alert Source Address* field.

e.g. lb1@loadbalancer.org

3. Enter an appropriate email address in the *Email Alert Destination Address* field.

e.g. alerts@loadbalancer.org

4. Click **Update**.


VIP Level Settings

 **Note** VIP level settings override the global settings.

Once configured, these settings apply to the individual VIP.

To configure VIP level email alerts:

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 4 Virtual Service* and click **Modify** next to the VIP to be configured.
2. Scroll down to the *Fallback Server* section.

Email Alert Destination Address	<input type="text" value="alerts@loadbalancer.org"/>	
---------------------------------	--	---

Cancel

Update

3. Enter an appropriate email address in the *Email Alert Destination Address* field.

e.g. alerts@loadbalancer.org

4. Click **Update**.

 **Note** You can set the *Email Alert Source Address* field as explained above if required to configure a default source address.

Configuring a Smart Host (SMTP relay)

For layer 4 services (and Heartbeat), email alerts are sent from the load balancer directly to the mail server defined in the destination domain's DNS MX record by default. Alternatively, a custom smart host (mail relay server) can be specified. A smart host is an email server through which approved devices can send emails. Where possible, we recommend that you use a smart host for email alerts as this often helps improve the deliverability of emails.

To configure a smart host:

1. Using the WebUI, navigate to: **Local Configuration > Physical - Advanced Configuration**.
2. Scroll down to the **SMTP Relay** section.
3. Specify the FQDN or IP address of the **Smart Host**.
4. Click **Update**.

Note

By default the **Smart Host** is set as the destination email domain's DNS MX record when the **Email Alert Destination Address** is configured - either at the global level or VIP level. It must either be left at its default setting or a custom smart host must be configured to enable email alerts to be sent.

Note





Layer 7 services do not use the smart host configured here. For layer 7, an SMTP server/smart host must be specified in the **eMail Server Address** field which is accessed using the WebUI menu option: **Cluster Configuration > Layer 7 Advanced Configuration**.

Layer 7

For layer 7 services, email settings are configured globally for all VIPs.

To configure global email alert settings for layer 7 services:

1. Using the WebUI, navigate to: **Cluster Configuration > Layer 7 Advanced Configuration**.

eMail Alert From	<input type="text" value="lb1@loadbalancer.org"/>	
eMail Alert To	<input type="text" value="alerts@loadbalancer.org"/>	
eMail Server Address	<input type="text" value="mail.loadbalancer.org"/>	
eMail Server Port	<input type="text" value="25"/>	

2. Enter an appropriate email address in the **eMail Alert From** field.

e.g. lb1@loadbalancer.org

3. Enter an appropriate email address in the **eMail Alert To** field.

e.g. alerts@loadbalancer.org

4. Enter an appropriate IP address/FQDN in the **eMail Server Address** field.

e.g. mail.loadbalancer.org

5. Enter an appropriate port in the **eMail Server Port** field.

e.g. 25

6. Click **Update**.







Real Server Monitoring & Control using the System Overview

Real Server Monitoring

The System Overview includes a visual display indicating the health status of all Virtual and Real Servers as shown in the example below:

System Overview ?



















2023-01-31 17:57:01 UTC

	VIRTUAL SERVICE	IP	PORTS	CONNS	PROTOCOL	METHOD	MODE	
	 HTTP-Cluster	192.168.110.150	80	0	HTTP	Layer 7	Proxy	
	 RDS-Cluster	192.168.110.152	3389	0	TCP	Layer 7	Proxy	

Clicking on each Virtual Service expands the view so that the associated Real Servers can also be seen:

System Overview ?

2023-01-31 17:57:20 UTC

	VIRTUAL SERVICE	IP	PORTS	CONNS	PROTOCOL	METHOD	MODE	
	 HTTP-Cluster	192.168.110.150	80	0	HTTP	Layer 7	Proxy	
	REAL SERVER	IP	PORTS	WEIGHT	CONNS			
	 IIS1	192.168.110.240	80	100	0	Drain	Halt	
	 IIS2	192.168.110.241	80	100	0	Drain	Halt	
	 RDS-Cluster	192.168.110.152	3389	0	TCP	Layer 7	Proxy	
	REAL SERVER	IP	PORTS	WEIGHT	CONNS			
	 RDS1	192.168.110.240	3389	100	0	Drain	Halt	
	 RDS2	192.168.110.241	3389	100	0	Drain	Halt	

The various colors used to indicate status are:



- **Green** - All Real Servers in the cluster are healthy
- **Yellow** - One or more Real Servers in the cluster has failed or has been taken offline using *Halt* or *Drain*
- **Red** - All Real Servers in the cluster have failed
- **Blue** - All Real Servers have been taken offline using *Drain* or *Halt* (see below)
- **Purple/Green** - Used to indicate that a particular VIP is used for HTTP to HTTPS redirection

This information is also displayed when clicking the system overview help button:

X

System Overview

The following colors and icons are used to show the real-time status of your Load balanced Virtual Services

Colour	Image	Details
Green	↑	Virtual Service / Real Server healthy
Yellow	⚠	Virtual Service needs attention
Blue	⚙	Real Server taken offline
Red	↓	Virtual Service / Real Server down
Purple	↑	Virtual Service FORCE-TO-HTTPS




The Virtual Services may be sorted using drag and drop, or by clicking on the column headings.

Real Server Control

The System Overview also enables each Real Server to be taken offline. This can be achieved in two ways:

1. **Drain** - This option allows all existing connections to complete & close gracefully. It also prevents any new connections being established.
2. **Halt** - This option drops all existing connections immediately without waiting. It also prevents any new connections being established.

The screen shot below shows that RDS2 has been placed in drain mode:

⚠	RDS-Cluster	192.168.110.152	3389	0	TCP	Layer 7	Proxy	
	<i>REAL SERVER</i>	<i>IP</i>	<i>PORTS</i>	<i>WEIGHT</i>	<i>CONNS</i>			
↑	<i>RDS1</i>	<i>192.168.110.240</i>	<i>3389</i>	<i>100</i>	<i>0</i>	<i>Drain</i>	<i>Halt</i>	
None	<i>RDS2</i>	<i>192.168.110.241</i>	<i>3389</i>	<i>0</i>	<i>0</i>	<i>Online (drain)</i>	<i>Halt</i>	

To bring RDS2 back online, click the **Online (drain)** link. If the server had been halted rather than drained, then the link would be displayed as **Online (Halt)**.

Note

If a particular Real Server is used in multiple VIPs you'll be asked if you'd like to apply the

offline/online action to all relevant VIPs or only a single VIP. This simplifies taking Real Servers offline for maintenance purposes.

Note

Halting or draining all Real Servers in a cluster activates the fallback server.

Ordering of VIPs

The display order of configured VIPs can be changed either by clicking on the column heading, or by drag and drop.

Sort by Column

If VIPs are ordered by a particular column, this is indicated using a single arrow next to the column heading as shown below:

System Overview ?

2023-01-31 18:00:40 UTC

	VIRTUAL SERVICE ▼	IP ↕	PORTS ↕	CONNS ↕	PROTOCOL ↕	METHOD ↕	MODE ↕	
↑	HTTP-Cluster	192.168.110.150	80	0	HTTP	Layer 7	Proxy	
⚠	RDS-Cluster	192.168.110.152	3389	0	TCP	Layer 7	Proxy	

In the example above, the VIPs are ordered alphanumerically by Virtual Service name in ascending order.

To change the order, click on the required column heading then click save. If you want to reverse the order for a particular column, click that column heading again.

For example, clicking on the IP column heading orders the VIPs by IP in descending order:

EDIT MODE

Cancel

Save

	VIRTUAL SERVICE ↕	IP ▲	PORTS ↕	CONNS ↕	PROTOCOL ↕	METHOD ↕	MODE ↕	
⚠	RDS-Cluster	192.168.110.152	3389	0	TCP	Layer 7	Proxy	
↑	HTTP-Cluster	192.168.110.150	80	0	HTTP	Layer 7	Proxy	

Clicking on the IP column heading again orders the VIPs by IP in ascending order:

EDIT MODE

Cancel

Save

	VIRTUAL SERVICE ↕	IP ▼	PORTS ↕	CONNS ↕	PROTOCOL ↕	METHOD ↕	MODE ↕	
↑	HTTP-Cluster	192.168.110.150	80	0	HTTP	Layer 7	Proxy	
⚠	RDS-Cluster	192.168.110.152	3389	0	TCP	Layer 7	Proxy	



Once you're happy with the ordering, click **Save**.




Drag & Drop

To order VIPs using drag and drop, simply click the mouse on any part of the VIP:

EDIT MODE

Cancel

Save




	VIRTUAL SERVICE	IP	PORTS	CONNS	PROTOCOL	METHOD	MODE
	RDS-Cluster 	192.168.110.152	3389	0	TCP	Layer 7	Proxy
	HTTP-Cluster	192.168.110.150	80	0	HTTP	Layer 7	Proxy

Then drag it to the required position:

EDIT MODE

Cancel

Save

	VIRTUAL SERVICE	IP	PORTS	CONNS	PROTOCOL	METHOD	MODE
	HTTP-Cluster	192.168.110.150	80	0	HTTP	Layer 7	Proxy
	RDS-Cluster 	192.168.110.152	3389	0	TCP	Layer 7	Proxy

And release it. Once you've set the required order, click **Save**.

Real Server Monitoring & Control using the HAProxy Statistics Report

Real Server Monitoring

The layer 7 Statistics Report shows the status of all layer 7 Virtual Services and Real Servers as shown in the example below:

Statistics Report for pid 8570

> General process information

```
pid = 8570 (process #1, nbproc = 1, nbthread = 1)
uptime = 0d 0h00m04s
system limits: memmax = unlimited; ulimit-n = 80049
maxsock = 80049; maxconn = 40000; maxpipes = 0
current conns = 1; current pipes = 0/0; conn rate = 2/sec
Running tasks: 1/26; idle = 100 %
```

active UP	backup UP
active UP, going down	backup UP, going down
active DOWN, going up	backup DOWN, going up
active or backup DOWN	not checked
active or backup DOWN for maintenance (MAINT)	
active or backup SOFT STOPPED for maintenance	

Note: "NOLB"/"DRAIN" = UP with load-balancing disabled

Display option:

- Scope :
- [Hide 'DOWN' servers](#)
- [Refresh now](#)
- [CSV export](#)

External resources:

- [Primary site](#)
- [Updates \(v1.8\)](#)
- [Online manual](#)

Web-Cluster																														
	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Status	Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr		Redis	LastChk	Wght	Act	Bok	Chk	Dwn	Dwntme	Thrtle
Frontend				0	0	-	0	0		40 000	0				0	0	0	0		0	OPEN									
backup	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	no check		1	-	Y					-
Web1	0	0	-	0	0		0	0	-	0	0	?	0	0				0	0	0	4s UP	L4OK in 0ms	100	Y	-	0	0	0s	-	
Web2	0	0	-	0	0		0	0	-	0	0	?	0	0				0	0	0	4s UP	L4OK in 0ms	100	Y	-	0	0	0s	-	
Web3	0	0	-	0	0		0	0	-	0	0	?	0	0				0	0	0	4s UP	L4OK in 0ms	100	Y	-	0	0	0s	-	
Backend	0	0		0	0		0	0	4 000	0	0	?	0	0	0	0		0	0	0	4s UP		300	3	1		0	0s		

ADS-S-Cluster																															
Queue			Session rate			Sessions					Bytes			Denied		Errors		Warnings		Server											
Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle		
Frontend			0	0	-	0	0	40 000	0	0						0	0	0	0	0	OPEN										
backup	0	0	-	0	0	0	0	-	0	0	?	0	0	0	0	0	0	0	0	0	no check		1	-	Y					-	
ADFS1	0	0	-	0	0	0	0	-	0	0	?	0	0	0	0	0	0	0	0	0	L7OK/200 in 0ms	100	Y	-	0	0	0s	-			
ADFS2	0	0	-	0	0	0	0	-	0	0	?	0	0	0	0	0	0	0	0	0	L7OK/200 in 0ms	100	Y	-	0	0	0s	-			
Backend	0	0	0	0	0	0	0	4 000	0	0	?	0	0	0	0	0	0	0	0	0	4s UP		200	2	1		0	0s			

Exchange-HTTP5																													
	Queue			Session rate			Sessions				Bytes		Denied		Req	Errors		Warnings		Status	Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out		Req	Resp	Conn	Resp		Retr	Redis	LastChk	Wght	Act	Bsk	Chk	Dwn	Dwntme
Frontend				0	0	-	0	0	40 000						0	0	0	0	0	0	OPEN								
bsbcpu	0	0	-	0	0	-	0	0	-	-	0	?	0	0	0	0	0	0	0	0	no check	1	-	Y					-
Exch1	0	0	-	0	0	-	0	0	-	-	0	?	0	0	0	0	0	0	0	0	4s UP	L4OK in 0ms	100	Y	-	0	0	0s	-
Exch2	0	0	-	0	0	-	0	0	-	-	0	?	0	0	0	0	0	0	0	0	4s UP	L4OK in 0ms	100	Y	-	0	0	0s	-
Backend	0	0	-	0	0	-	0	0	4 000	0	0	?	0	0	0	0	0	0	0	0	4s UP		200	2	1	0	0	0s	

	stats																														
	Queue			Session rate			Sessions						Bytes				Denied		Errors			Warnings		Server							
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Refr	Redis	Status	LastChk	Wght	Act	Bok	Chk	Dwn	Dwntme	Thrtle	
Frontend				2	2	-	1	2	2 000	2			488	262	0	0	0					OPEN									
Backend	0	0		0	0				200	0	0s		488	262	0	0		0	0	0	0	4s UP		0	0	0	0		0		

To access the page, navigate to: *Reports > Layer 7 Status* - a new tabbed window will be opened.



 **Note**

From v8.8.1, all statistics are automatically refreshed every 30 seconds by default. This can be disabled if preferred or the refresh interval can be changed. To configure these settings, use the *Auto-refresh Stats Page* and *Auto-refresh Interval* options available under *Cluster Configuration > Layer 7 - Advanced Configuration*.

Real Server Control

It's also possible to control layer 7 Real Servers using the HAProxy statistics Report. By default this is not enabled.

To enable Real Server control:

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Advanced*.
2. Enable (check) the *Advanced Stats* option.
3. Click **Update**.
4. Reload HAProxy using the button at the top of screen. With this setting, the HAProxy stats page has the ability to control the state of Real Servers as shown below:

> General process information

pid = 7354 (process #1, nbproc = 1, nbthread = 1)
 uptime = 0d 0h0m29s
 system limits: memmax = unlimited; ulimit-n = 80049
 maxsock = 80049; maxconn = 40000; maxpipes = 0
 current conns = 1; current pipes = 0/0; conn rate = 5/sec
 Running tasks: 1/28; idle = 100 %

active UP
 active UP, going down
 active DOWN, going up
 active or backup DOWN
 active or backup DOWN for maintenance (MAINT)
 active or backup SOFT STOPPED for maintenance
 backup UP
 backup UP, going down
 backup DOWN, going up
 not checked
 Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option:

- Scope:
- [Hide DOWN servers](#)
- [Refresh now](#)
- [CSV export](#)

External resources:

- [Primary page](#)
- [Updates \(v1.8\)](#)
- [Online manual](#)

■	Web-Cluster																				Server									
	Queue			Session rate			Sessions					Bytes		Denied		Errors		Warnings		Status	LastChk	Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp			Retr	Redis	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend																						OPEN								
<input type="checkbox"/> backup	0	0	-	0	0	-	0	0	40 000	0	0	?	0	0	0	0	0	0	0	0	0	no check							-	
<input type="checkbox"/> Web1	0	0	-	0	0	-	0	0	0	0	?	0	0	0	0	0	0	0	0	0	29s UP	L4OK in 0ms	100	Y	-	0	0	0s	-	
<input type="checkbox"/> Web2	0	0	-	0	0	-	0	0	0	0	?	0	0	0	0	0	0	0	0	0	29s UP	L4OK in 0ms	100	Y	-	0	0	0s	-	
<input type="checkbox"/> Web3	0	0	-	0	0	-	0	0	0	0	?	0	0	0	0	0	0	0	0	0	29s UP	L4OK in 0ms	100	Y	-	0	0	0s	-	
<input type="checkbox"/> Backend	0	0	-	0	0	-	0	0	4 000	0	0	?	0	0	0	0	0	0	0	0	29s UP		300	3	1		0	0s		

Choose the action to perform on the checked servers:

Apply

ADFS-Cluster										Set state to READY Set state to DRAIN Set state to MAINT Health: disable checks Health: enable checks Health: force UP Health: force HOLD Health: force DOWN Agent: disable checks														
	Queue			Session rate			LbTot	Bytes		Denied	Req	Conn	Resp	Retr	Redis	Status	LastChk	Server						
	Cur	Max	Limit	Cur	Max	Limit		Cur	In	Out	Req	Resp	Conn	Resp	Retr			Redis	Wght	Act	Bck	Chk	Dwn	Downtm
Frontend	0	0	-	0	0	-	0	0	0	0	0	0	0	0	0	OPEN								
backup	0	0	-	0	0	-	0	?	0	0	0	0	0	0	0	no check			1	-	Y			-
ADFS1	0	0	-	0	0	-	0	?	0	0	0	0	0	0	0	29s UP	L7OK/200 in 0ms	100	Y	-	0	0	0s	-
ADFS2	0	0	-	0	0	-	0	?	0	0	0	0	0	0	0	29s UP	L7OK/200 in 0ms	100	Y	-	0	0	0s	-
Backend	0	0	-	0	0	-	0	?	0	0	0	0	0	0	0	29s UP		200	2	1		0	0s	

Choose the action to perform on the checked servers:

Apply

Exchange-HTTPS															Server										
	Queue			Session rate			Cur	Bytes		Denied	Req	Conn	Resp	Retr	Redis	Status	LastChk	Server							
	Cur	Max	Limit	Cur	Max	Limit		Last	In	Out	Req	Resp	Req	Conn	Resp	Retr		Redis	Wght	Act	Bck	Chk	Dwn	Downtme	Thrftle
Frontend	0	0	-	0	0	-	0	0	40 000	0	0	0	0	0	0	0	OPEN								
backup	0	0	-	0	0	-	0	0	0	0	0	0	0	0	0	0	no check								
Exch1	0	0	-	0	0	-	0	0	0	0	0	0	0	0	0	0	29s UP	L4OK in 0ms	100	Y	-	0	0	0s	-
Exch2	0	0	-	0	0	-	0	0	0	0	0	0	0	0	0	0	29s UP	L4OK in 0ms	100	Y	-	0	0	0s	-
Backend	0	0	-	0	0	-	0	0	4 000	0	0	0	0	0	0	0	200 2s UP		200	2	1		0	0s	

Choose the action to perform on the checked servers:

Apply

- Use the checkboxes to select the relevant Real Server(s), then select the required action in the dropdown, then click **Apply**.

Chapter 9 - Appliance Clustering for HA

Introduction

Loadbalancer.org appliances can be deployed as single unit or as a clustered pair.

Note

We always recommend deploying a clustered pair to avoid introducing a single point of failure.

Clustered Pair Concepts

When configured as a clustered pair, the appliances work in **Active-Passive** mode. In this mode the active appliance (normally the Primary) handles all traffic under normal circumstances. If the active appliance fails, the passive appliance (normally the Secondary) becomes active and handles all traffic.

Whilst HA clustering can be configured at any time, we generally recommend that the Primary appliance should be fully configured first and then the Secondary appliance should be added to create the clustered pair. Once this is done, all load balanced services configured on the Primary are automatically replicated to the Secondary. Pairing must be performed on the appliance that is to be the Primary appliance.

Note

For Enterprise Azure, the HA pair should be configured first. In Azure, when creating a VIP using an HA pair, two private IPs must be specified – the first for the VIP when it's active on the Primary and the other for the VIP when it's active on the Secondary. Configuring the HA pair first, enables both IPs to be specified when the VIP is created.

Primary/Secondary Operation

Pair Communication

To allow the pair to function correctly, the Primary and Secondary must be able to:

- Perform an ICMP echo request (ping) to each other
- Communicate with each other on TCP port 22 (or the selected custom port if the default SSH port has been changed)
- Communicate with each other on UDP port 6694 (or the selected custom port if the default heartbeat port has been changed)
- Communicate with each other on TCP port 9443 (or the selected custom port if the default WebUI port has been changed)

Note

For details on setting custom ports for SSH and the WebUI, please refer to [Service Socket Addresses](#), for details on setting a custom port for heartbeat, please refer to [Configuring Heartbeat](#).

Heartbeat

By default, heartbeat uses ucast on UDP port 6694 to communicate between the Primary and Secondary appliances. The link enables the state of each to be monitored by the other and permits a failover to the passive



appliance if the active appliance should fail. For hardware appliances, it's possible to configure both ucast and serial communication if required.

Note

For hardware appliances, if the load balancer pair is located in close proximity, enabling serial communication in addition to ucast is recommended. Once the serial cable is connected between the appliances, serial comms can be enabled using the WebUI menu option: **Cluster Configuration > Heartbeat Configuration**.

Ping checks to a common node such as the default gateway can also be configured. If the active node loses access to the ping node, the system will fail-over to the peer. However, if both nodes lose access, no fail-over will occur.

Primary Secondary Replication

Once the Primary and Secondary are paired, all load balanced services are automatically replicated from Primary to Secondary. This ensures that should the Primary appliance fail, the Secondary is already configured to run the same services. Replication of the configured load balanced services from the Primary to the Secondary appliance occurs over the network using SSH/SCP.

Settings that are NOT Replicated to the Secondary Appliance

Local appliance settings such as hostname and DNS, IP address configuration and time/date settings are not replicated and must be manually configured on each appliance. The table below lists all settings by WebUI menu option that are not replicated from the Primary appliance to the Secondary appliance when an HA pair is configured.

WebUI Main Menu Option	Sub Menu Option	Description
Local Configuration	Hostname & DNS	Hostname and DNS settings
Local Configuration	Network Interface Configuration	Interface IP addresses, bonding configuration and VLANs
Local Configuration	Routing	Default gateways and static routes
Local Configuration	System Date & time	Time and date related settings
Local Configuration	Physical – Advanced Configuration	Various appliance settings
Local Configuration	Portal Management	Portal management settings
Local Configuration	Security	Security settings
Local Configuration	SNMP Configuration	SNMP settings
Local Configuration	Graphing	Graphing settings
Local Configuration	License Key	Appliance licensing
Maintenance	Backup & Restore	Local XML backups
Maintenance	Software Updates	Appliance software updates
Maintenance	Fallback Page	Fallback page configuration

WebUI Main Menu Option	Sub Menu Option	Description
Maintenance	Firewall Script	Firewall (iptables) configuration
Maintenance	Firewall Lockdown Wizard	Appliance management lockdown settings

Manually Forcing Appliance Synchronization

If changes are made to the Primary appliance of an HA Pair whilst the Secondary appliance is down, the HA Pair must be manually re-synchronised once the secondary is back up.

To force Primary to Secondary synchronization:

1. Using the WebUI on the Primary, navigate to: **Maintenance > Backup & Restore**.
2. Select the Synchronisation tab.
3. Click **Synchronize Configuration with peer**.


To Create an HA Pair (Add a Secondary)

Note

If you have already run the firewall lockdown wizard on either appliance, you'll need to ensure that it is temporarily disabled on both appliances whilst performing the pairing process.

1. Power up a second appliance that will be the Secondary and configure initial network settings - for more details on initial deployment and network setup, please refer to [Chapter 4 - Appliance Fundamentals](#).
2. Using the WebUI of the Primary appliance, navigate to: **Cluster Configuration > High-Availability Configuration**.

Create a Clustered Pair

 **LOADBALANCER**

Local IP address

192.168.110.40

IP address of new peer

192.168.110.41

Password for loadbalancer user on peer

••••••••••



Add new node

3. Specify the IP address and the "loadbalancer" user's password for the Secondary (peer) appliance as shown above.
4. Click **Add new node**.
5. A warning will be displayed indicating that the pairing process will overwrite the new Secondary appliance's

existing configuration, click **OK** to continue.



- The pairing process now commences as shown below:

Create a Clustered Pair

 LOADBALANCER Primary IP: 192.168.110.40	Local IP address 192.168.110.40
Attempting to pair..	
 LOADBALANCER Secondary IP: 192.168.110.41	IP address of new peer 192.168.110.41
	Password for loadbalancer user on peer ●●●●●●●●
	configuring



- Once complete, the following will be displayed under: *Cluster Configuration > High Availability Configuration* on the Primary appliance:

High Availability Configuration - primary


 LOADBALANCER Primary IP: 192.168.110.40	Break Clustered Pair
 LOADBALANCER Secondary IP: 192.168.110.41	

The following will be displayed under: *Cluster Configuration > High Availability Configuration* on the Secondary appliance:

High Availability Configuration - secondary

 LOADBALANCER Secondary IP: 192.168.110.41	Break Clustered Pair
	Promote to Primary
 LOADBALANCER Primary IP: 192.168.110.40	

- Now restart heartbeat as prompted in the "Commit changes" message box at the top of the screen.

 **Note** Clicking the **Restart Heartbeat** button on the Primary appliance will also automatically

restart heartbeat on the Secondary appliance.

To Break an HA Pair (Remove a Secondary)

1. Using the WebUI of the Primary or Secondary appliance, navigate to: *Cluster Configuration > High-Availability Configuration* (shows the Primary appliance).

High Availability Configuration - primary

The screenshot displays the 'High Availability Configuration - primary' interface. On the left, there are two appliance cards. The top card is for the 'Primary' appliance, labeled 'LOADBALANCER' with a status icon, and shows the IP address 'IP: 192.168.110.40'. The bottom card is for the 'Secondary' appliance, also labeled 'LOADBALANCER', and shows the IP address 'IP: 192.168.110.41'. On the right side of the interface, there is a prominent red button with the text 'Break Clustered Pair'.


2. To break the pair, click the **Break Clustered Pair** button.
3. Click **OK** to confirm you want to proceed.

High Availability Configuration - primary

This screenshot shows the same 'High Availability Configuration - primary' interface, but in a transitional state. The red 'Break Clustered Pair' button is now a disabled, light-colored button. Between the Primary and Secondary appliance cards, there is a loading spinner icon and the text 'Attempting to break...'. The appliance details, including the 'LOADBALANCER' labels and IP addresses (192.168.110.40 for Primary and 192.168.110.41 for Secondary), remain visible.

4. Once the process is complete, the pairing configuration screen will be displayed on both appliances:

Create a Clustered Pair

 **LOADBALANCER**

Local IP address

IP address of new peer

Password for loadbalancer user on peer

Add new node

5. To complete the reconfiguration, restart the system services on both appliances as directed in the "Commit changes" message box.

Notes

1. Load balanced services will be momentarily interrupted as system services are restarted.
2. After the pair is broken, the Secondary appliance will be left configured as a Secondary and any configured load balanced services will remain.
3. If you later want to use the Secondary as a Primary, use the **Cluster Configuration > High Availability Configuration** menu option on the Secondary to setup a new pair. The Secondary will then be re-configured as a Primary, and the added peer will be configured as a Secondary.
4. If you want to reset the configuration to factory defaults, use the WebUI menu option: **Maintenance > Backup & Restore > Restore > Restore Manufacturer's Defaults**.


Promoting a Secondary to Primary


This is useful if the Primary appliance fails and you'd like to change the now active Secondary to be a Primary, and then add the repaired/replaced Primary back as a Secondary appliance.

To promote a Secondary appliance to a Primary:

1. Using the WebUI of the Secondary appliance, navigate to: **Cluster Configuration > High-Availability Configuration**.

High Availability Configuration - secondary

 **LOADBALANCER** **Secondary**
IP: 192.168.110.41

 **LOADBALANCER** **Primary**
IP: 192.168.110.40


Break Clustered Pair

Promote to Primary





2. Click **Promote**.
3. Click **OK** to confirm you want to proceed.

High Availability Configuration - secondary

 **LOADBALANCER** Secondary

IP: 192.168.110.41


Promoting Secondary Node...

 **LOADBALANCER** Primary


IP: 192.168.110.40

Break Clustered Pair

Promote to Primary

 **Note** If the Primary is still up and operational, it will not be possible to promote the Secondary.

4. Once complete, the appliance will be configured as a Primary.


 **Note** Please refer to [Chapter 14 - Backup & Restore and Disaster Recovery](#) for more details on how to recover from various appliance failure scenarios.

Configuring Heartbeat

The default heartbeat settings are appropriate in most cases, but can be changed if required.

To configure heartbeat:

1. Using the WebUI of the Primary appliance, navigate to: **Cluster Configuration > Heartbeat Configuration**.

 **Note** The screen shot below shows the configuration screen for a hardware appliance. The virtual appliance does not have the serial option checkbox in the communications method section.

Heartbeat Configuration

Communication method		
Serial	<input type="checkbox"/>	?
UDP Unicast	<input checked="" type="checkbox"/>	?
UDP Broadcast (<i>Deprecated</i>)	Off ▾	?
UDP Port for broadcast & unicast	6694	?

Peer Failure Detection		
Keep-alive message interval	3 seconds	?
Dead peer timer	10 seconds	?
Warning timer	5 seconds	?

Routing Failure Detection		
Test IP addresses		?
Test time-out	10 seconds	?

Email Alerts		
Email Alert Destination Address		?
Email Alert Source Address		?

Automatic Fail-back		
	<input type="checkbox"/>	?

Modify Heartbeat configuration

Communication Method

Serial - Enable or disable heartbeat Primary/Secondary communication over the serial port (hardware appliances only). Ucast is the default heartbeat communication method. However, if the load balancer pair is located in close proximity, enabling serial communication in addition to ucast is recommended. This method requires a null modem cable (supplied with the appliance) to be connected to the serial port of each appliance.

Note

Console access via the serial port is not supported. The serial port can only be used for heartbeat communication between appliances.

UDP Unicast - Enable or disable unicast heartbeat Primary/Secondary communication. This is the default method of heartbeat communication and uses unicast UDP between Primary and Secondary, with a destination port specified by the *UDP Port for broadcast & unicast* parameter. When unicast is enabled, the load balancer determines the correct interface and IP addresses to use based upon the configured Secondary IP address.

UDP Broadcast (Deprecated) - Enable or disable broadcast heartbeat Primary/Secondary communication, and choose the interface. This option is deprecated - please migrate to Unicast. This method of heartbeat communication uses broadcast UDP between Primary and Secondary, with a destination port specified by the *UDP Port for broadcast & unicast* parameter. Care must be taken when using broadcast on multiple pairs of load



balancers in the same network. Each high-availability pair must operate on a different UDP port if they are not to interfere with each other. If heartbeat communication over the network is required, it is recommended that unicast be used in preference to broadcast.

UDP Port for unicast & broadcast - The UDP port number used by heartbeat for network communication over unicast or broadcast. By default, heartbeat uses UDP port 6694 for unicast or broadcast communication. If you have multiple load balancer pairs on the same subnet, and wish to use broadcast, you will need to set each pair to a different UDP port.

Peer Failure Detection

Keep-alive message interval - Specify the number of seconds between keepalive pings. The Keepalive setting must be less than the warntime and deadtime.

Dead peer timer - The number of seconds communication can fail before a fail over is performed. A very low setting of deadtime could cause unexpected failovers.

Warning timer - If communication fails for this length of time write a warning to the logs. This is useful for tuning your deadtime without causing failovers in production.

Routing Failure Detection

Test IP address - Specify one or more mutually accessible IP address to test network availability. A good ping node to specify is the IP address of a router that both the Primary and Secondary node can access. If the active node loses access to the ping node, the system will fail-over to the peer. However, if both nodes lose access, no fail-over will occur. Multiple IP addresses may be given, separated by spaces or commas. In this case, all addresses must be reachable for the routing test to pass.

Test time-out - Specify the time-out, in seconds, for the routing test. If a response is not received from the test address within the time-out period, the route to that host will be considered dead.

Email Alerts

Email Alert Destination Address - Specify the email address where to send heartbeat alerts. In the event of failover or failback the email address specified will receive an alert.

Email Alert Source Address - Specify the email address from where to send heartbeat alerts. In the event of failover failback the email specified will send an alert.

Note

Both Primary and Secondary appliances will send an email in the event of a failover or failback.

Fail-back Settings

Automatic Fail-back - Enable/disable auto-failback. This option controls the cluster behavior when the Primary returns to service after a failure. With Automatic Fail-back enabled, the Primary will automatically return to active status, taking back the floating IP addresses from the Secondary. With Automatic Fail-back disabled, the Secondary will remain active and will retain the floating IP addresses. Fail-over back to the Primary must then be controlled manually.

Note

Automatic Fail-back is disabled by default. Manual intervention is required to force the repaired Primary to become active and the Secondary appliance to return to passive mode. For more information on controlling failback, please refer to [Forcing Primary/Secondary Failover & Failback](#).

Configuring a Smart Host (SMTP relay)

For Heartbeat (and layer 4 services), email alerts are sent from the load balancer directly to the mail server defined in the destination domain's DNS MX record by default. Alternatively, a custom smart host (mail relay server) can be specified. A smart host is an email server through which approved devices can send emails. Where possible, we recommend that you use a smart host for email alerts as this often helps improve the deliverability of emails.

To configure a Smart Host:

1. Using the WebUI, navigate to: *Local Configuration > Physical - Advanced Configuration*.
2. Scroll down to the *SMTP Relay* section.
3. Specify the FQDN or IP address of the *Smart Host*.
4. Click **Update**.

Note

By default the *Smart Host* is set as the destination email domain's DNS MX record when the *Email Alert Destination Address* is configured. It must either be left at its default setting or a custom smart host must be configured to enable email alerts to be sent.

Connection State & Persistence Table Replication

Layer 4 VIPs

If you want the current connection state and persistence table to be available when the active appliance (typically the Primary) fails over to the passive appliance (typically the Secondary), then you can start the synchronization daemons on both appliances to replicate this data in real time as detailed below.

First, login to the Primary appliance as "root" and run the following commands:

```
ipvsadm --start-daemon master
ipvsadm --start-daemon backup
```

Next, login to the Secondary appliance as "root" and run the following commands:

```
ipvsadm --start-daemon master
ipvsadm --start-daemon backup
```

Note

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, configure the required option(s) and click **Update**.



To ensure that these sync daemons are started on each reboot, put these commands in the file `/etc/rc.d/rc.firewall`. This can be done using the WebUI menu option: **Maintenance > Firewall Script**. Make sure that the full path is specified in the firewall script, i.e.

```
/usr/local/sbin/ipvsadm --start-daemon master
/usr/local/sbin/ipvsadm --start-daemon backup
```

After a few seconds you can confirm that it is working by viewing the connections report on each appliance which is available in the WebUI by navigating to: **Reports > Layer 4 Current Connections** as shown in the following examples:

The active appliance:

```
IPVS connection entries
pro expire state      source          virtual         destination
TCP 02:13  NONE              192.168.64.7:0  192.168.111.221:23 192.168.110.240:23
TCP 12:12  ESTABLISHED      192.168.64.7:53177 192.168.111.221:23 192.168.110.240:23
TCP 12:14  ESTABLISHED      192.168.64.7:53180 192.168.111.221:23 192.168.110.240:23
```

The passive appliance:

```
IPVS connection entries
pro expire state      source          virtual         destination
TCP 12:08  ESTABLISHED      192.168.64.7:53177 192.168.111.221:23 192.168.110.240:23
TCP 02:12  NONE              192.168.64.7:0    192.168.111.221:23 192.168.110.240:23
TCP 12:12  ESTABLISHED      192.168.64.7:53180 192.168.111.221:23 192.168.110.240:23
```

You can also run the following command at the command line:

```
ipvsadm -Lc
```

As shown, the state of all current connections as well as the persistence entries (i.e. those in state "NONE") are replicated to the passive device. This enables current connections to continue through the passive appliance should the active appliance fail.

To stop the replication, run the following commands on both appliances:

```
ipvsadm --stop-daemon master
ipvsadm --stop-daemon backup
```

Note

Setting this option can generate a high level traffic between the Primary and Secondary appliances.

Note

Once configured, you'll see multicast UDP traffic from the active appliance to multicast IP

address 224.0.0.81 on port 8848.

Layer 7 VIPs

If you want the current persistence tables to be available when the active appliance (typically the Primary) fails over to the passive appliance (typically the Secondary), then persistence table replication must be enabled. Enabling this option will replicate all persistence tables for layer 7 VIPs that use stick table based persistence.

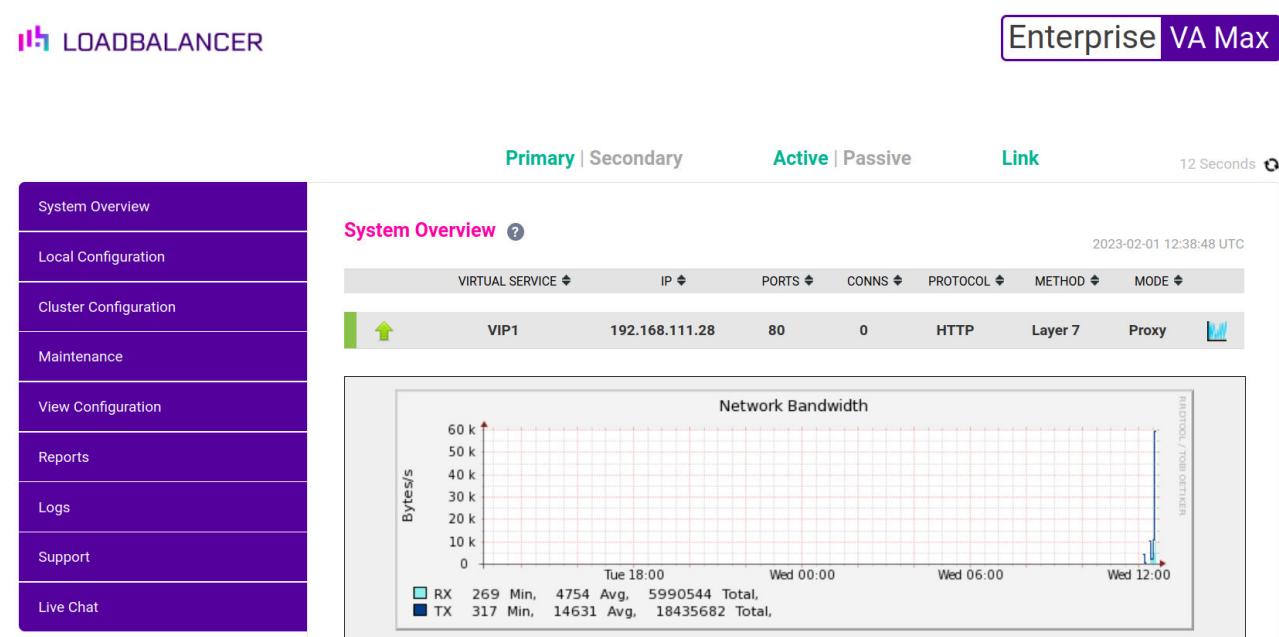
To enable persistence state table replication:

1. Using the WebUI, navigate to: **Cluster Configuration > Layer 7 - Advanced Configuration**.
2. Enable the **Persistence Table Replication**.
3. Click **Update**.
4. Now reload HAProxy using the **Reload** button in the "Commit changes" message box at the top of the screen.

Clustered Pair Diagnostics

Heartbeat State Diagnostics

The status of the appliance is shown at the top of the screen. For a HA pair, the normal status for the Primary appliance is shown below:



This shows that:

- This is the Primary appliance and it's currently active
- The heartbeat link between the Primary & Secondary is up and working correctly

If no VIPs are configured, the status on the Primary & Secondary appear as follows:

Primary | Secondary

Active | Passive

Link

Primary | Secondary

Active | Passive

Link

Other possible states:

Primary Secondary	Active Passive	Link	this is a Primary appliance, it's active, no Secondary appliance has been configured.
Primary Secondary	Active Passive	Link	this is a Primary appliance, it's active, a Secondary has been configured but the link to the Secondary is down. Action: check & verify the heartbeat configuration & if required restart heartbeat on both appliances.
Primary Secondary	Active Passive	Link	this is a Secondary appliance, it's active (a failover from the Primary has occurred) and the heartbeat link to the Primary has been established.
Primary Secondary	Active Passive	Link	this is a Primary appliance, a Secondary appliance has been configured, but the link is down (e.g. serial cable unplugged) so the state cannot be determined. In this case the floating IPs may be active on both appliances. Action: check & verify the heartbeat configuration, check the serial cable (if applicable), check heartbeat logs & if required restart heartbeat on both appliances.

Split Brain Scenarios

Split brain can occur if heartbeat on the Primary/Secondary clustered pair can no longer communicate with one another. In this case both appliances will assume that the other appliance is down and will bring up the Virtual Services. The system status will look similar to the following on both appliances:

Primary | Secondary

Active | Passive

Link

55 Seconds

Error: The heartbeat link to the secondary node is down

Error: heartbeat may be active on both load balancers

When heartbeat communication is re-established, heartbeat will automatically attempt to resolve the split brain and ensure that only one of the appliances is active. If heartbeat fails to do this automatically, the system status will show as follows on both appliances:



Primary | Secondary

Active | Passive

Link

12 Seconds

Error: heartbeat may be active on both load balancers. To force this node to take control, click the button below.

Take over

The **Take over** button can then be used on either Primary or Secondary to attempt to force that appliance to become active.

Forcing Primary/Secondary Failover & Failback

Under normal conditions, the Primary is the active appliance and the Secondary is the passive appliance. If required, a failover can be triggered so the Secondary becomes active and the Primary passive. This can be acheived in 3 ways as detailed below. Once the Secondary is active, the same methods can be used on the Primary to force failback.

Method 1 - Using the Take over button on the Secondary

1. Using the WebUI on the Secondary, select: *System Overview*.
2. Click **[Advanced]** in the blue message box.

System Overview

2023-02-01 13:11:18 UTC

Information: This device is currently passive. Please see the active device for Virtual Service statistics.
[Advanced]

Take over Make this node active

3. Click **Take Over**.

Method 2 - Using the Make Active button on the Secondary

1. Using the WebUI on the Secondary, navigate to: *Cluster Configuration > High Availability Configuration*.

High Availability Configuration - secondary

LOADBALANCER

Secondary

IP: 192.168.110.41

LOADBALANCER

Primary

IP: 192.168.110.40

Break Clustered Pair

Make Active

Promote to Primary

2. Click **Make Active**.

Method 3 - Using the hb_takeover command

Run the following command on the Secondary appliance:

```
/usr/local/sbin/hb_takeover.php all
```

Note

This command can either be run on the console, via an SSH session or via the WebUI menu option: *Local Configuration > Execute shell command*.

Note

The "Execute Shell Command" menu option is disabled by default. This can be enabled using the WebUI option: *Local Configuration > Security*. Set *Appliance Security Mode* to **Custom** then click **Update**.

Note

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, configure the required option(s) and click **Update**.

Testing & Verifying Primary/Secondary Replication & Failover

Important

Make sure you verify that Primary/Secondary failover occurs correctly before going live. This proves the resilience of the HA cluster and makes you aware of the failover/failback process.

Note

When testing appliance fail-over, if heartbeat is configured to use only the serial link, don't just pull the serial cable out. This will not cause a fail-over but will cause a split brain (i.e. both appliances active) to occur. Testing must be done by pulling both the network and serial cable (if used) as detailed below.

STEP 1 - Verify Basic Settings for the Clustered Pair

1. On the Primary appliance verify that the system status appears as follows:

Primary | Secondary **Active** | Passive Link

2. On the Secondary appliance verify that the system status appears as follows:

Primary | **Secondary** Active | **Passive** Link

STEP 2 - Verify Replication

1. Verify that the load balanced services have been replicated to the Secondary appliance, this can be done by using either the *View Configuration* or *Edit Configuration* menus to validate that the same Virtual & Real Servers exist on the Secondary as on the Primary.



STEP 3 - Verify Failover to the Secondary (Using the Take Over Button)

1. On the Secondary appliance, click the **[Advanced]** option in the information box, then click the **Take Over** button.
2. Verify that the Secondary's status changes to *Active*:

Primary | **Secondary** **Active** | Passive [Link](#)

3. Verify that the Primary's status changes to *Passive*:

Primary | **Secondary** **Active** | **Passive** [Link](#)

4. Using the WebUI menu option: *View Configuration > Network Configuration*, verify that the floating IPs associated with the VIPs have been brought up on the Secondary appliance and brought down on the Primary.

e.g. the partial screen shot below from the View Network Configuration screen on the Secondary appliance shows the status of **eth0**:

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:0c:29:92:18:4f brd ff:ff:ff:ff:ff:ff
    inet 192.168.111.223/18 brd 192.168.127.255 scope global eth0
        valid_lft forever preferred_lft forever
    inet 192.168.111.72/18 brd 192.168.127.255 scope global secondary eth0
        valid_lft forever preferred_lft forever
```

This shows the secondary IP address 192.168.111.72 (the VIP address) is up and therefore the Secondary has become active as intended.

STEP 4 - Verify Fail-back to the Primary (Using the Take Over Button)

1. On the Primary appliance, click the **[Advanced]** option in the information box, then click the **Take Over** button.
2. Verify that the Primary's status has changed to *Active*:

Primary | **Secondary** **Active** | Passive [Link](#)

3. Verify that the Secondary's status has changed to *Passive*:

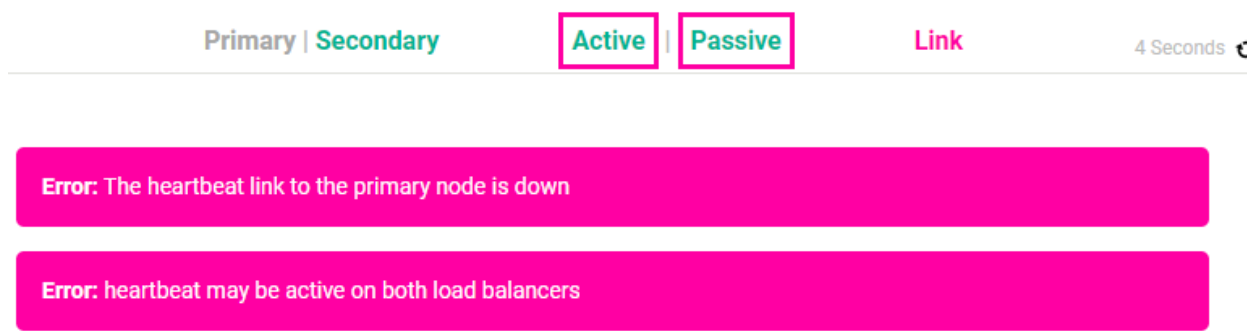
Primary | **Secondary** **Active** | **Passive** [Link](#)

4. Also, using the WebUI menu option: *View Configuration > Network Configuration*, verify that the floating IPs associated with the VIPs have been brought up on the Primary appliance and brought down on the Secondary (see STEP 3 above for more details).

STEP 5 - Verify Failover to the Secondary (when Removing the Network and Serial Cables from the Primary)



1. Remove the network cable and serial cable (if applicable) from the Primary.
2. Verify that the Secondary's status has changed as follows:



This indicates that the Secondary is unable to communicate with the Primary. This means that either the Primary is down, or is still up but is unreachable. In both cases the Secondary will become active.

3. On the Secondary using the WebUI menu option: **View Configuration > Network Configuration**, verify that the floating IPs associated with the VIPs have been brought up (see STEP 3 above for more details).

STEP 6 - Verify Normal Operation Resumes (when Reconnecting the Network & Serial cables to the Primary)

1. Reconnect the cables to the Primary.
2. Verify that the Primary's status is set to **Active**:

Primary | Secondary Active | Passive Link

3. Verify that the Secondary has changed to **Passive**:

Primary | Secondary Active | Passive Link

4. Also, using the WebUI menu option: **View Configuration > Network Configuration**, verify that the floating IPs associated with the VIPs have been brought up on the Primary appliance and down on the Secondary.

Note

If the power cable on the Primary is removed rather than disconnecting the network cable and serial cable (if applicable), when the Primary is brought back online it assumes the passive role and the Secondary remains active. The **Take over** button on the Primary can then be used to make the Primary active. If **Auto Fail-Back** is enabled, the Primary automatically assumes the active role when brought back online.

Heartbeat Log

The log provides a useful insight into the health of an HA pair and can be viewed using the WebUI menu option: **Logs > Heartbeat**. The log can be searched using the search box at the top of the screen or can be downloaded for external analysis. The log can be refreshed using the **Check Status** button. The log file is located here: /var/log/ha.log.

Late Heartbeats

Late Heartbeats are caused when the heartbeat packets either arrive at the other appliance later than the **Warning Timer** configured in the WebUI under **Cluster Configuration > Heartbeat Configuration** or not at all.

The following log snippet shows three entries reporting a late heartbeat:

```
Apr 26 21:03:53 lbmaster heartbeat: [2474]: WARN: Late heartbeat: Node lbmaster: interval 5110 ms
Apr 26 21:04:30 lbmaster heartbeat: [2474]: WARN: Late heartbeat: Node lbmaster: interval 6430 ms
Apr 26 21:04:35 lbmaster heartbeat: [2474]: WARN: Late heartbeat: Node lbmaster: interval 5830 ms
```

This suggests that there is either high latency or packet loss and is the first signs of issues that can lead to a split brain.

Troubleshooting late Heartbeats

- Verify that the system load average is acceptable - An overloaded system can be struggling to send or receive messages.
- Check if the network card is saturated - An overloaded NIC can add latency.



Note

For more information, please refer to [Resource Utilization Graphs](#).

- Confirm that there are no bottlenecks or packet loss further down the wire - Such as links between data centres where there may be traffic congestion/contention.
- Once a possible cause has been identified and addressed, monitor the situation to make sure the issue has been resolved.

Chapter 10 - Configuration Examples

Introduction

This section presents a number of simple examples that illustrate how the appliance can be configured.

Initial Network Settings & WebUI Access

The examples below assume that the [Network Setup Wizard](#) has already been run and an appropriate IP address and network mask have been assigned to **eth0**. For details on how to access the WebUI, please refer to [Accessing the WebUI](#).

Example 1 - One-Arm DR Mode (Single Appliance)

This web server example uses a DR mode Virtual Service (VIP) with two Real Servers (RIPs). DR mode offers the highest possible performance because return traffic passes directly from the Real Servers to the client bypassing the load balancer. For more information, please refer to [Layer 4 DR Mode](#).


Configuration Overview


- **Verify Network Settings** - A single Interface is required and was configured during the Network Setup Wizard. This can be verified using the WebUI.
- **Configure the Virtual Service (VIP)** - The Virtual Service specifies the IP address and port presented to clients, the forwarding method to use - in this example layer 4 DR Mode and other settings such as health checks and timeouts.
- **Configure the Real Servers (RIPs)** - The Real Servers are the load balanced backend servers.
- **Configure the Real Backend Servers for DR Mode** - For DR mode to work correctly, the "ARP Problem" must be solved on each Real Server.


Verify Network Settings


1. Using the WebUI, navigate to: *Local Configuration > Network Interface Configuration*.
2. Scroll down to the *IP Address Assignment* section.

IP Address Assignment


eth0


eth1


eth2


eth3

eth0

192.168.2.120/24

MTU

1500

 bytes

3. Verify that the IP address & subnet mask assigned during the Network Setup Wizard are configured, e.g. **192.168.2.120/24**.



Note

Typically **eth0** is used for single-arm configurations although this is not mandatory.

Configure the Virtual Service (VIP)

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 4 - Virtual Services* and click **Add a new Virtual Service**.

Virtual Service		
Label	<input type="text" value="ExVIP1"/>	
IP Address	<input type="text" value="192.168.2.150"/>	
Ports	<input type="text" value="80"/>	
Protocol		
Protocol	<input type="text" value="TCP"/>	
Forwarding		
Forwarding Method	<input type="text" value="Direct Routing"/>	
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

2. Enter a suitable Label (name) for the VIP, e.g. **ExVIP1**.
3. Enter a valid IP address, e.g. **192.168.2.150**.
4. Enter a valid port, e.g. **80**.
5. Ensure that the *Protocol* is set to **TCP**.
6. Ensure that the *Forwarding Method* is set to **Direct Routing**.
7. Click **Update**.

Configure the Real Servers (RIPs)

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 4 - Real Servers* and click **Add a new Real Server** next to the Virtual Service created above.

Label	<input type="text" value="RIP1"/>	
Real Server IP Address	<input type="text" value="192.168.2.151"/>	
Weight	<input type="text" value="100"/>	
Minimum Connections	<input type="text" value="0"/>	
Maximum Connections	<input type="text" value="0"/>	
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>



2. Enter a suitable Label (name) for the RIP, e.g. **RIP1**.
3. Enter a valid IP address, e.g. **192.168.2.151**.

 **Note**

There is no option to specify a Real Server port because port redirection is not possible when using DR mode. Traffic will be passed to the Real Servers on the same port that it was received on at the Virtual Service.

4. The weight can be left at the default value of 100. This is a relative number, so if all Real Servers have the same value each server will receive the same amount of traffic.
5. Leave **Minimum Connections** & **Maximum Connections** set to 0, this means unrestricted.
6. Click **Update**.
7. Now repeat these steps to add the second Real Server.

Configure the Real Backend Servers for DR Mode

The steps required to solve the ARP problem depend on the particular Real Server OS used. For more information on the "ARP Problem" and the configuration requirements for various operating systems, please refer to [DR Mode Considerations](#).

 **Note**

Failure to correctly configure the Real Servers to handle the "ARP Problem" is the most common issue when using DR mode.

Basic Testing & Verification

Once configured, a few quick checks can be performed to verify the setup:

1. Using the **System Overview** in the WebUI, verify that the VIP & RIPs are shown as active (green).
2. Using a browser, navigate to the VIP address, i.e. **http://192.168.2.150** to verify that you can reach the Real Servers via the Virtual Service.
3. Check **Reports > Layer 4 Current Connections** to ensure that client connections are reported in state "ESTABLISHED". If connections are in state "SYN_RECV", this normally indicates that the "ARP Problem" on the Real Servers has not been solved.

Example 2 - Two-Arm NAT Mode (Clustered Pair)

This web server example uses a layer 4 NAT mode Virtual Service (VIP) with two Real Servers (RIPs). NAT mode is not as fast as DR mode, but Real Server configuration is simpler since NAT mode only requires that the default gateway on each Real Server is set to be the load balancer. For more information, please refer to [Layer 4 NAT Mode](#).

This example also demonstrates how to configure a clustered pair using two appliances.

 **Note**

Using two appliances configured as a clustered pair is Loadbalancer.org's recommended configuration and ensures that no single point of failure is introduced.

Note

When using a clustered pair with NAT mode, a floating IP address should be used as the default gateway. This allows the gateway to move between appliances should a failover occur.

Configuration Overview

- **Configure the Primary's Network Settings** - two interfaces are required, in this example **eth0** which was configured during the network setup wizard is used as the **internal** (Real Server) interface and **eth1** is used as the **external** (client) interface.
- **Configure the Secondary's Network Settings** - two interfaces are required, in this example **eth0** which was configured during the network setup wizard is used as the **internal** (Real Server) interface and **eth1** is used as the **external** (client) interface.

Note





Alternatively, two VLANs could be created on a single adapter.

- **Configure the Virtual Service (VIP) using the Primary's WebUI** - The Virtual Service specifies the IP address and port presented to clients, the forwarding method to use - in this example layer 4 NAT Mode and other settings such as health checks and timeouts.
- **Configure the Real Servers (RIPs) using the Primary's WebUI** - The Real Servers are the load balanced backend servers.
- **Create the HA Clustered Pair using the Primary's WebUI** - Pair the Primary & Secondary appliances to create an HA clustered pair.
- **Configure the Real Backend Servers for NAT Mode** - For NAT mode to work correctly, the default gateway on each Real Server must be set to be an IP on the load balancer.

Configure the Primary's Network Settings

1. Using the WebUI on the Primary, navigate to: *Local Configuration > Network Interface Configuration*.
2. Scroll down to the *IP Address Assignment* section.

IP Address Assignment

				
	eth0	eth1	eth2	eth3
eth0	<input type="text" value="192.168.2.120/24"/>			MTU <input type="text" value="1500"/> bytes
eth1	<input type="text" value="192.168.20.120/24"/>			MTU <input type="text" value="1500"/> bytes

3. The IP address & subnet mask will already be set for **eth0**, this was configured during the Network Setup.
4. Specify an IP address & mask for **eth1**, e.g. **192.168.20.120/24**.

Note


Make sure that both interfaces are connected to the relevant switch.


5. Click **Configure Interfaces**.


Configure the Secondary's Network Settings


1. Using the WebUI on the Secondary, navigate to: *Local Configuration > Network Interface Configuration*.

IP Address Assignment


eth0


eth1


eth2


eth3

eth0

192.168.2.121/24

eth1

192.168.20.121/24

MTU

1500

bytes

MTU

1500

bytes

2. The IP address & subnet mask will already be set for **eth0**, this was configured during the Network Setup.
3. Specify an IP address & mask for **eth1**, e.g. **192.168.20.121/24**.

Note

Make sure that both interfaces are connected to the relevant switch.

4. Click **Configure Interfaces**.

Configure the Virtual Service (VIP) using the Primary's WebUI

1. Using the WebUI of the Primary appliance, navigate to: *Cluster Configuration > Layer 4 - Virtual Services* and click **Add a new Virtual Service**.

Virtual Service		
Label	<input type="text" value="ExVIP2"/>	?
IP Address	<input type="text" value="192.168.20.150"/>	?
Ports	<input type="text" value="80"/>	?
Protocol		
Protocol	<input type="text" value="TCP"/>	?
Forwarding		
Forwarding Method	<input type="text" value="NAT"/>	?
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

2. Enter a suitable label (name) for the VIP, e.g. **ExVIP2**.
3. Enter a valid IP address, e.g. **192.168.20.150**.
4. Enter a valid port, e.g. **80**.
5. Set the *Forwarding Method* to **NAT**.
6. Click **Update**.

Configure the Real Servers (RIPs) using the Primary's WebUI

Add the Real Servers, i.e. the web servers. This should be done on the Primary appliance.

1. Using the WebUI of the Primary appliance, navigate to: *Cluster Configuration > Layer 4 - Real Servers* and click **Add a new Real Server**.

Label	<input type="text" value="RIP1"/>	?
Real Server IP Address	<input type="text" value="192.168.2.151"/>	?
Real Server Port	<input type="text" value="80"/>	?
Weight	<input type="text" value="100"/>	?
Minimum Connections	<input type="text" value="0"/>	?
Maximum Connections	<input type="text" value="0"/>	?
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

2. Enter a suitable Label (name) for the RIP, e.g. **RIP1**.
3. Enter a valid IP address, e.g. **192.168.2.151**.
4. Enter a valid port, e.g. **80**.
5. The weight can be left at the default value of 100. This is a relative number, so if all Real Servers have the

same value each server will receive the same amount of traffic.

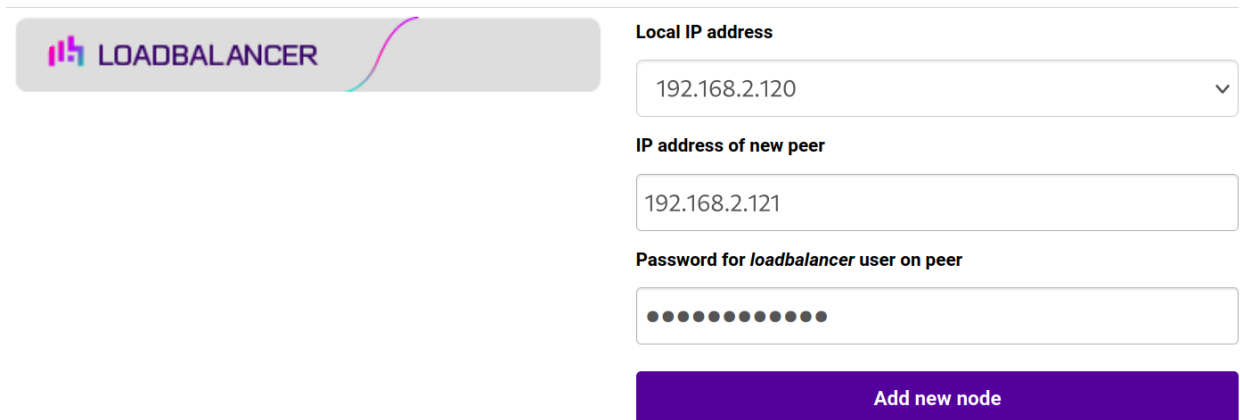
6. Leave *Minimum Connections* & *Maximum Connections* set to 0, this means unrestricted.
7. Click **Update**.
8. Now repeat these steps to add the second Real Server.

Create the HA Clustered Pair using the Primary's WebUI

This will configure the HA Pair and also replicate the Virtual Service and Real Servers to the Secondary appliance. The Secondary appliance will then be kept in-sync with the Primary. This must be performed on the Primary appliance.

1. Using the WebUI of the Primary appliance, navigate to: *Cluster Configuration* > *High Availability Configuration*

Create a Clustered Pair



LOADBALANCER

Local IP address

192.168.2.120

IP address of new peer

192.168.2.121


Password for *loadbalancer* user on peer

●●●●●●●●●●●●

Add new node


2. Leave the *Local IP address* set as the address assigned to **eth0**, in this case **192.168.2.120**.
3. Specify the *IP address of new peer* (i.e. the Secondary appliance), in this case **192.168.2.121**.
4. Specify the "loadbalancer" user's password for the Secondary appliance.
5. Click **Add new node**.
6. A warning will be displayed indicating that the pairing process will overwrite the new Secondary appliance's existing configuration, click **OK** to continue.
7. The pairing process will start as shown below:


Create a Clustered Pair

 **LOADBALANCER**

Primary

IP: 192.168.2.120


Attempting to pair..

 **LOADBALANCER**

Secondary

IP: 192.168.2.121

Local IP address

192.168.2.120

IP address of new peer

192.168.2.121


Password for *loadbalancer* user on peer

••••••••••

configuring


8. Once completed successfully, the following will be shown:

High Availability Configuration - primary

 **LOADBALANCER**

Primary

IP: 192.168.2.120

 **LOADBALANCER**

Secondary

IP: 192.168.2.121

Break Clustered Pair

Make Active

9. To finalize the configuration, click **Restart Heartbeat** in the "Commit changes" message box at the top of the screen.

Review Heartbeat Settings

- Using the WebUI on the Primary appliance, navigate to: *Cluster Configuration > Heartbeat Configuration*.
- The default Heartbeat settings are normally fine for most situations. For details of all heartbeat options, please refer to [Configuring Heartbeat](#).

Checking the Status

A successfully configured clustered pair will display the following status on the Primary appliance:

Primary | Secondary **Active** | Passive [Link](#)

And the following status on the Secondary appliance:

Primary | **Secondary** Active | **Passive** [Link](#)

Verify Replication to the Secondary



To verify that the new VIP & RIPs have been replicated correctly, open the WebUI on the Secondary appliance and navigate to: **Cluster Configuration > Layer 4 - Virtual Services** and **Cluster Configuration > Layer 4 - Real Servers** and verify that the VIPs and RIPs have been replicated successfully and their configuration is the same as on the Primary appliance.

Configure the Real Backend Servers for NAT Mode

When using NAT mode, each Real Server's default gateway must be set to be an IP address on the load balancer. For a clustered pair, a floating IP address must be used. This allows the gateway address to move to the Secondary should a failover occur. The floating IP address must be added on the Primary appliance.

1. Using the WebUI of the Primary appliance, navigate to: **Cluster Configuration > Floating IPs**.

New Floating IP

192.168.2.254

Add Floating IP

1. Specify the IP address you'd like to use for the default gateway, e.g. **192.168.2.254**.
2. Click **Add Floating IP**.

Now configure the default gateway on each Real Server to use this address.

Basic Testing & Verification

A few quick checks can be performed to verify the configuration:

1. On the Primary, use **System Overview** to check that the VIP & RIPs are shown as active (green).
2. Using a browser, navigate to the VIP address, i.e. **http://192.168.2.150** to verify that you can reach the Real Servers via the Virtual Service.
3. On the Primary, check **Reports > Layer 4 Current Connections** to ensure that client connections are reported in state "ESTABLISHED". If not and instead "SYN-RECV" is shown, double-check that you have set the default gateway on all Real Servers to be the floating IP address on the load balancer.
4. On

Example 3 - One-Arm SNAT Mode & SSL Termination (Single Appliance)

In this example, STunnel is used to terminate SSL on the load balancer. STunnel passes the unencrypted HTTP traffic to an HAProxy VIP which then load balances the traffic between two Real Servers. HAProxy does not offer the raw throughput of layer 4, but is still a high performance solution.

In this example it's assumed that the Real Server application has not been designed to track & share session details between Real Servers. Therefore, cookie based persistence is enabled on the load balancer to ensure that clients connect to the same Real Server on each subsequent connection (within the persistence timeout window). If persistence is not configured, new connections may get distributed to a different Real Server which may break the application.

Layer 7 SNAT mode does not require any mode specific changes to the Real Servers.



Note

SSL termination on the load balancer can be very CPU intensive. In most cases, for a scalable solution, terminating SSL on the Real Servers is the best option.





Configuration Overview

- **Verify Network Settings** - A single Interface is required and was configured during the Network Setup Wizard. This can be verified using the WebUI.
- **Configure the Virtual Service (VIP)** - The Virtual Service specifies the IP address and port presented to clients, the forwarding method to use - in this example layer 4 DR Mode and other settings such as health checks and timeouts.
- **Configure the Real Servers (RIPs)** - The Real Servers are the load balanced backend servers.
- **Configure SSL Termination** - STunnel is used by default for SSL terminations.

Verify Network Settings

1. Using the WebUI, navigate to: *Local Configuration > Network Interface Configuration*.

IP Address Assignment

eth0

192.168.2.120/24

eth1

eth2

eth3

eth0

MTU 1500 bytes

2. Specify the IP address & mask for **eth0** - normally **eth0** is used for one-arm configurations although this is not mandatory, e.g. **192.168.2.120/24**.
3. Click **Configure Interfaces**.
4. Using the WebUI, navigate to: *Local Configuration > DNS & Hostname*.
5. Specify the DNS server(s).

Domain Name Server	Primary	192.168.2.254	?
	Secondary		?
	Tertiary		?

6. Click **Update**.
7. Using the WebUI, navigate to: *Local Configuration > Routing*.



Default Gateway			
IP v4	<input type="text" value="192.168.2.254"/>	via interface	auto ▼ ?
IP v6	<input type="text"/>	via interface	auto ▼ ?

8. Specify the Default Gateway, e.g. 192.168.2.254.

9. Click **Configure Routing**.

Configure the Virtual Service (VIP)

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Virtual Services* and click **Add a new Virtual Service**.

Virtual Service		[Advanced +]
Label	<input type="text" value="ExVIP3"/>	?
IP Address	<input type="text" value="192.168.2.150"/>	?
Ports	<input type="text" value="80"/>	?
Protocol		[Advanced +]
Layer 7 Protocol	HTTP Mode ▼	?

Cancel
Update

2. Enter a suitable Label (name) for the VIP, e.g. **ExVIP3**.

3. Enter a valid IP address, e.g. **192.168.2.150**.

4. Enter a valid port, e.g. **80**.

5. Leave *Layer 7 Protocol* set to **HTTP Mode**.

6. Click **Update**.

Configure the Real Servers (RIPs)

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Real Servers* and click **Add a new Real Server**.

Label	<input type="text" value="RIP1"/>	?
Real Server IP Address	<input type="text" value="192.168.2.151"/>	?
Real Server Port	<input type="text" value="80"/>	?
Re-Encrypt to Backend	<input type="checkbox"/>	?
Enable Redirect	<input type="checkbox"/>	?
Weight	<input type="text" value="100"/>	?

Cancel
Update

2. Enter a suitable Label (name) for the RIP, e.g. **RIP1**.
3. Enter a valid IP address, e.g. **192.168.2.151**.
4. Enter a valid port, e.g. **80**.
5. The **Weight** defaults to 100 making Real Servers active as soon as HAProxy is restarted.
6. Click **Update**.
7. Now repeat these steps to add the second Real Server.
8. Reload HAProxy to apply the new settings using the link provided in the "Commit changes" message box at the top of the screen.

Configure SSL Termination

1. Using the WebUI, navigate to: *Cluster Configuration > SSL Termination* and click **Add a new Virtual Service**.

Label	<input type="text" value="SSL-ExVIP3"/>	?
Associated Virtual Service	<input type="text" value="ExVIP3"/>	?
Virtual Service Port	<input type="text" value="443"/>	?
SSL Operation Mode	<input type="text" value="High Security"/>	?
SSL Certificate	<input type="text" value="Default Self Signed Certificate"/>	?
Source IP Address	<input type="text"/>	?
Enable Proxy Protocol	<input checked="" type="checkbox"/>	?
Bind Proxy Protocol to L7 VIP	<input type="text" value="ExVIP3"/>	?

Cancel
Update

2. Set *Associated Virtual Service* to the Layer 7 VIP created earlier, e.g. **ExVIP3**.

Note The label will be automatically set to **SSL-ExVIP3**. This can be edited if required.

3. Leave **Virtual Service Port** set to **443**.
4. Leave the other settings at their default values.
5. Click **Update**.
6. Reload STunnel and HAProxy to apply the new settings using the buttons in "Commit changes" message box.

Note

When creating the SSL Virtual Service, by default a self-signed certificate is used. This is ideal for testing but needs to be replaced for live deployments. Certificates can be added using the WebUI option: **Cluster Configuration > SSL Certificate**. Once added, these will appear in the **SSL Certificate** dropdown when creating the SSL VIP.

Note

For more information on configuring SSL termination, please refer to [SSL Termination](#).

Basic Testing & Verification

A few quick checks can be performed to verify the configuration:

1. Using **System Overview**, verify that the VIP & RIP are shown as active (green).
2. Using a browser, navigate to the VIP address, i.e. **http://192.168.2.150** to verify that you can reach the Real Servers via the Virtual Service using HTTP.
3. Using a browser, navigate to the STunnel SSL VIP address, i.e. **https://192.168.2.150** to verify that you can reach the Real Servers via the Virtual Service using HTTPS.

Example 4 - Load Balancing FTP

This section provides guidance on how FTP can be load balanced using both layer 4 and layer 7 Virtual Services.

FTP is a multi-port service in both active and passive modes:

Active 20 (Data), 21 (Command)

Passive 21 (Command), High Port (Data)

Layer 4

To load balance FTP at layer 4, configure a layer 4 VIP and set the **Ports** field to 21. The load balancer will then automatically configure the Virtual Service with the additional ports that are required for both active and passive mode. The full port list is:

```
Port 20 - active mode data channel
Port 21 - active and passive mode command channel
ports 1024 to 65535 - passive mode data channel
```

Note

The full passive port range is configured to allow for any passive range on the FTP servers.

Layer 7

Active Mode

In active mode, the FTP server connects back to the client to establish the data channel, so it must be aware of the client's IP address. To achieve this, TProxy must be enabled to make the load balancer transparent at layer 7. For this to work, two subnets must be used - the Virtual Service in one subnet and Real Servers (i.e. the FTP servers) in the other and the default gateway on each FTP server must be the load balancer. For more details on TProxy, please refer to [Transparency at Layer 7](#).

Also, to ensure that the client receives a connection from the same address that it established the control connection to, an iptables SNAT rule for each FTP server must be configured in the load balancer's firewall script. The format of the required rule is as follows:

```
iptables -t nat -A POSTROUTING -p tcp -s <FTP-Server-IP> -j SNAT --to-source <FTP-VIP>
```

e.g.

```
iptables -t nat -A POSTROUTING -p tcp -s 10.20.1.1 -j SNAT --to-source 192.168.2.180
```

Make sure you add one rule for each FTP server in the cluster.

Note

These rules can be added to the firewall script using the WebUI menu option: **Maintenance > Firewall Script**.

Active Mode - Key Points:

- Use separate subnets for the VIP & RIPv
- Enable TProxy for the layer 7 VIP
- Set the default gateway on the FTP servers to be an IP on the load balancer (For an HA pair, this should be a floating IP to permit failover to the Secondary appliance)
- The VIP must be configured to listen on port 21
- Ensure the Layer 7 Protocol is set to TCP Mode
- Increase the default client & server HAProxy timeouts to 5 minutes
- Add the SNAT firewall rules - one for each FTP server

Windows Example

1. Create a L7 VIP with the following settings. Change the name and IP address as required:

Layer 7 - Add a new Virtual Service

Virtual Service		[Advanced +]
Label	<input type="text" value="FTP-ClusterACTV"/>	?
IP Address	<input type="text" value="192.168.2.150"/>	?
Ports	<input type="text" value="21"/>	?
Protocol		
Layer 7 Protocol	<input type="text" value="TCP Mode"/>	?
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

2. Click **Update** to create the VIP.
3. Click **Modify** next to the newly created VIP.
4. Scroll down to the *Other* section and click **[Advanced]**.
5. Enable (check) the *Timeout* checkbox then set both *Client Timeout* and *Server Timeout* to **5m** (5 minutes).
6. Define the FTP servers as RIPs as illustrated below (these must be on a different subnet to the VIP to enable TProxy to work correctly):

Layer 7 Add a new Real Server - FTP-ClusterACTV

Label	<input type="text" value="ftp1"/>	?
Real Server IP Address	<input type="text" value="10.10.1.1"/>	?
Real Server Port	<input type="text" value="21"/>	?
Re-Encrypt to Backend	<input type="checkbox"/>	?
Weight	<input type="text" value="100"/>	?
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

7. Enable TProxy for the VIP - modify the VIP, scroll down to the *Other* section, click **[Advanced]** and enable (check) *Transparent Proxy*.
8. Now reload HAProxy using the **Reload** button in the "Commit changes" message box at the top of the screen.
9. Define a SNAT rule for each FTP server using the WebUI menu option: **Maintenance > Firewall Script**.

e.g.

```
iptables -t nat -A POSTROUTING -p tcp -s 10.10.1.1 -j SNAT --to-source 192.168.2.150
iptables -t nat -A POSTROUTING -p tcp -s 10.10.1.2 -j SNAT --to-source 192.168.2.150
```

10. Configure the default gateway on each FTP server to be the load balancer. For an HAP pair this should be a floating IP address to allow it to float (move) between the Primary & Secondary appliance. This can be added using the WebUI menu option: **Cluster Configuration > Floating IPs**.
11. Active FTP clients should now be able to connect to the VIP address (192.168.2.150) and view the directory listing successfully.

Passive Mode

In passive mode all connections are initiated by the client. The server passes the client a port to use for the inbound data connection. By default, FTP servers can use a wide range of ports for the inbound connection and it's often useful to limit this range. For more information on configuring passive mode for a range of OSs, please refer to [Configuring FTP Passive Mode Port Range](#).

Passive Mode - Key Points:

- It's sensible to use a controlled passive port range, this can be configured on the FTP server
- Configure the VIP to listen on port 21 and also the same passive range configured on the FTP servers, e.g. 50000-50100
- Configure the RIPv **without** specifying a port
- Ensure the Layer 7 Protocol is set to TCP Mode
- Increase the default client & server HAProxy timeouts to 5 minutes
- If transparency is required (for passive mode this is optional), enable TProxy. To enable TProxy, modify the VIP, scroll down to the **Other** section, click **[Advanced]** and enable (check) **Transparent Proxy**.

Note

If TProxy is enabled, certain topology requirements must be met. Also the default gateway on each FTP server must be set to be an IP on the load balancer. For an HA pair, this should be a floating IP to allow failover to the Secondary appliance. For More information on using TProxy, please refer to [Transparency at Layer 7](#).

- Set the Client Timeout & Real Server Timeout to 5m (i.e. 5 minutes)
- Set the Persistence Mode to Source IP
- To ensure that the correct address is passed back to the client, configure the external address on each FTP server to be the VIP address.
 - for Windows 2008 R2 & later use the **External IP address of Firewall** field
 - for Linux vsftpd use the directive: **pasv_address=xxx.xxx.xxx.xxx**
 - for Linux ProFTPD use the directive: **MasqueradeAddress=xxx.xxx.xxx.xxx**

Windows Example

1. Create a layer VIP with the following settings. Change the name, IP address & passive port range as required:

Layer 7 - Add a new Virtual Service

Virtual Service		[Advanced +]
Label	<input type="text" value="FTP-ClusterPASV"/>	?
IP Address	<input type="text" value="192.168.2.150"/>	?
Ports	<input type="text" value="21,50000-50100"/>	?
Protocol		
Layer 7 Protocol	<input type="text" value="TCP Mode"/>	?
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

2. Configure the VIP to listen on both the control port (21) and passive data range (e.g. 50000-50100) as shown.
3. Click **Update** to create the VIP.
4. Click **Modify** next to the newly created VIP.
5. Scroll down to the *Other* section and click **[Advanced]**.
6. Enable (check) the *Timeout* checkbox then set both *Client Timeout* and *Server Timeout* to **5m** (5 minutes).
7. Define the FTP servers as RIPS leaving the port field blank as illustrated below:

Layer 7 Add a new Real Server - FTP-ClusterPASV

Label	<input type="text" value="ftp1"/>	?
Real Server IP Address	<input type="text" value="10.10.1.1"/>	?
Real Server Port	<input type="text"/>	?
Re-Encrypt to Backend	<input type="checkbox"/>	?
Weight	<input type="text" value="100"/>	?
		<input type="button" value="Cancel"/> <input type="button" value="Update"/>

8. Now reload HAProxy using the **Reload** button in the "Commit changes" message box at the top of the screen.
9. On each FTP server, open IIS Manager and double click the *FTP Firewall Support* icon. Ensure that the same passive port range is specified and set the *External IP Address of Firewall* to be the same as the Virtual Service (VIP) address as shown in the example below:



FTP Firewall Support

The settings on this page let you configure your FTP server to accept passive connections from an external firewall.

Data Channel Port Range:

Example: 5000-6000

External IP Address of Firewall:

Example: 10.0.0.1



Note

This ensures that the VIP address is passed back to the client to use for the subsequent inbound connection.

10. If TProxy is enabled, make sure the gateway of each FTP server is set to be an IP on the load balancer. For an HA pair, this should be a floating IP to permit failover to the Secondary appliance.
11. Now restart both IIS **and** the Microsoft FTP Service on each FTP server or reboot each server.
12. Passive FTP clients should now be able to connect to the VIP address (192.168.2.180) and view the directory listing successfully.

Configuring FTP Passive Mode Port Range

Limiting passive ports allows your firewall to be more tightly locked down.

For Windows 2008 R2 & Later

As mentioned in the example above, the IIS Management console can be used to configure the passive port range for Windows.

Specify a suitable port range, in the example below this is 50000-50100.



FTP Firewall Support

The settings on this page let you configure your FTP server to accept passive connections from an external firewall.

Data Channel Port Range:

Example: 5000-6000

External IP Address of Firewall:

Example: 10.0.0.1

For Linux



- For **vsftpd**, the following line can be added to the vsftpd.conf file to limit the port range:

```
pasv_max_port - max is 65535  
pasv_min_port - min is 1024
```

- For **proftpd**, the following line can be added to the proftpd.conf file to limit the port range:

```
PassivePorts 50000 - 50100
```

- For **pureftpd**, the following startup switch can be used:

```
-p --passiveportrange <min port:max port>
```

Chapter 11 - Diagnostics & Troubleshooting

Note

Don't hesitate to contact support@loadbalancer.org if you're experiencing any issues or need assistance configuring your appliance.

The Appliance

If your appliance is exhibiting performance issues or you're having administrative related problems, the resource utilization Graphs, administration log and the Apache logs may offer further insight.

Resource Utilization

The graphs in the System Overview can be used to give a quick overview of appliance resource utilization. By default, the system Overview includes graphs for **Network Bandwidth**, **System Load Average** and **Memory Usage**. For more information, please refer to [Resource Utilization Graphs](#).

Administration Log

The appliance maintains a log of all administration tasks and all notifications. The log can be viewed using the WebUI menu option: **Logs > Load Balancer**. The log can be searched using the search box at the top of the screen or can be downloaded for external analysis. The log can be refreshed using the **Check Status** button. The log file is located here: `/var/log/lbadmin.log`.

Apache Logs

Apache is used to run the WebUI. The following log files may be useful when diagnosing WebUI related issues:

Apache Access Log

This log can be viewed using the WebUI menu option: **Logs > Apache**. The log can be searched using the search box at the top of the screen or can be downloaded for external analysis. The log can be refreshed using the **Check Status** button. The log file is located here: `/var/log/httpd/user_access.log`.

Apache Error Log

This log can be viewed using the WebUI menu option: **Logs > Apache Error**. The log can be searched using the search box at the top of the screen or can be downloaded for external analysis. The log can be refreshed using the **Check Status** button. The log file is located here: `/var/log/httpd/error.log`.

Layer 4 & Layer 7 Virtual Services

Virtual services may not function as expected for a variety of reasons. This section suggests various diagnostic methods that can help identify problems and also looks at potential configuration issues that could be the cause.

Checking Service State using the System Overview

The system overview can be used to quickly check the state of layer 4 and layer 7 Virtual Services and their associated Real Servers.

Note

for more information, please also refer to [Real Server Monitoring & Control using the System Overview](#).



The first screenshot shows that the Virtual Service **Web-Cluster** is healthy and that all associated Real Servers are up.

System Overview ?

2023-02-01 15:50:53 UTC

	VIRTUAL SERVICE	IP	PORTS	CONNS	PROTOCOL	METHOD	MODE	
	Web-Cluster	192.168.110.235	80	0	HTTP	Layer 7	Proxy	
	REAL SERVER	IP	PORTS	WEIGHT	CONNS			
	Web1	192.168.110.240	80	100	0	Drain	Halt	
	Web2	192.168.110.241	80	100	0	Drain	Halt	
	Web3	192.168.110.242	80	100	0	Drain	Halt	

The second screenshot shows that the Virtual Service is now colored yellow and marked with an exclamation mark indicating that one or more of the associated Real Servers is not available. The colored tab and the arrow on the left show the current status of each Real Server.

System Overview ?

2023-02-01 15:52:21 UTC

	VIRTUAL SERVICE	IP	PORTS	CONNS	PROTOCOL	METHOD	MODE	
	Web-Cluster	192.168.110.235	80	0	HTTP	Layer 7	Proxy	
	REAL SERVER	IP	PORTS	WEIGHT	CONNS			
	Web1	192.168.110.240	80	100	0	Drain	Halt	
	Web2	192.168.110.241	80	100	0	Online (halt)		
	Web3	192.168.110.242	80	100	0	Drain	Halt	

In the example above:

- **Web-Cluster** is yellow indicating that one or more of the Real Servers in the cluster is down - either due to a failed health check or because it's been manually taken offline (either drained or halted)
- **Web1** is green, this indicates that it's passing it's health check
- **Web2** is blue, this indicates that it has been either Halted or Drained, in this example Halt has been used as indicated by Online (Halt) being displayed. If it had been drained it would show as Online (Drain)
- **Web3** is red, this indicates that it has failed it's health check

In addition to the above colours, Purple/Green is used to indicate that a particular VIP is used for HTTP to HTTPS redirection.

VIP(s) Fail to appear in the System Overview

If you have configured new layer 7 VIPs and these have not automatically appeared in the System Overview, make sure that you've reloaded or restarted HAProxy.

VIPs & RIPs are Green but Users Can't Connect

If you've configured your VIPs and RIPs and everything looks fine (green) in the System Overview but users still cannot connect, there are a number of causes for this, some are dependent on whether you've configured layer 4 or layer 7 VIPs.

General

Multi-port VIPs - Does your VIP listen on multiple ports, for example, **80,443**? if so, make sure that the Real Server port field is left blank so that traffic is passed through on the correct port. If you specify for example **80**, all traffic including traffic intended to reach port 443 will be passed to port 80.

Note

DR mode does not have a Real Server port field since port translation is not possible when using this mode. Traffic is always passed through to the same port as received by the VIP.

Layer 7 VIPs

Protocol - Have you configured the correct **Layer 7 Protocol**? The default protocol for new layer 7 VIPs is HTTP. This is fine for web based traffic typically on port 80, but if you've configured your layer 7 VIP to load balance something else such as HTTPS on port 443 or RDP on port 3389 then you'll need to change the **Layer 7 Protocol** dropdown to TCP.

TProxy - If you've enabled Transparent Proxy, there are certain topology and other requirements that must be met. For more information about enabling transparency at layer 7, please refer to [Transparency at Layer 7](#).

Layer 4 VIPs

Layer 4 DR mode and NAT mode have certain configuration requirements. It's important to remember that the health checks performed by the load balancer verify that the **load balancer** can successfully access the server/service/application. This does not verify that each server has been configured correctly to enable **client** access. The following sections explain how the connection state can be used to determine if the Real Servers have been configured correctly, and also what are the configuration requirements for each mode.

DR Mode

Connection State - Use the WebUI option: **Reports > Layer 4 Current Connections** to view the current traffic in detail. Connections that are in the state **SYN_RECV** can indicate that the "ARP Problem" has not been correctly solved on the associated Real Server.

Real Server Configuration Requirements - For layer 4 DR mode VIPs, the "ARP Problem" must be solved on all associated Real Servers - the exact steps required depend on the particular Real Server OS. For more information, please refer to [DR Mode Considerations](#).

NAT Mode

Connection State - Use the WebUI option: **Reports > Layer 4 Current Connections** to view the current traffic in detail. Any connections in state **SYN_RECV** can indicate that the default gateway on the associated Real Server has not been set to be an IP address on the load balancer.

Real Server Configuration Requirements - For layer 4 NAT mode VIPs, the default gateway on all associated Real Server must be configured to be an IP address on the load balancer to ensure that client return traffic passes back via the load balancer. For an HA pair, this should be a floating IP address to allow the IP address to float



(move) between the Primary and Secondary.

Diagnosing Real Server Issues

If Real Servers are down (red) in the System Overview, this means that the configured health check is failing which can be caused by a variety of reason.

- Verify that the application/service is actually running on each Real Server.
- Is the health check correctly configured and is it appropriate for the Real Servers? The default check for TCP services is a simple port connect - if this has been changed to a negotiate HTTP health check for example, has a valid Request & Response been configured? For more information on configuring health checks, please refer to [Chapter 8 - Real Server Health Monitoring & Control](#).
- Check that you can ping the Real Server from the load balancer. This can be done using the WebUI option: **Local Configuration > Execute Shell Command**, at the console or via an SSH session, e.g.

```
ping -c 4 192.168.111.240
```

The **-c 4** causes 4 ping attempts, and the command then ends. This is important when running the command from the WebUI to ensure it terminates cleanly.

Note

The "Execute Shell Command" menu option is disabled by default. This can be enabled using the WebUI option: **Local Configuration > Security**. Set **Appliance Security Mode** to **Custom** then click **Update**.

Note

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: **Local Configuration > Security**. You'll need to Set **Appliance Security Mode** to **Custom**, configure the required option(s) and click **Update**.

- Verify that you can connect to the application port from the load balancer. This can be done using telnet at the console or via an SSH session:

```
telnet 192.168.111.240 <application Port>
```

e.g. for a web server listening on port 80:

This example shows that a telnet connection was successfully established:

```
[root@lbmaster ~]# telnet 192.168.110.240 80
Trying 192.168.110.240...
Connected to 192.168.110.240.
Escape character is '^['.
```

This example shows that the telnet connection failed:



```
[root@lbmaster ~]# telnet 192.168.110.240 80
Trying 192.168.110.240...
telnet: connect to address 192.168.110.240: Connection refused
```



Note

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI menu option: **Local Configuration > Security**. You'll need to Set **Appliance Security Mode** to **Custom**, configure the required option(s) and click **Update**.

You can also use nmap on the load balancer to verify that the required Real Server port is open:

```
[root@lbmaster ~]# nmap 192.168.110.241

Starting Nmap 5.51 ( http://nmap.org ) at 2022-07-12 10:54 UTC
Nmap scan report for 192.168.110.240
Host is up (0.0012s latency).
Not shown: 996 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
443/tcp   open  https
3389/tcp  open  ms-term-serv
MAC Address: 00:50:56:82:0B:D3 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 4.97 seconds
```

Another option is curl:

```
[root@lbmaster ~]# curl http://192.168.110.241
or
[root@lbmaster ~]# curl http://host.mydomain.com
```

For a working web server listening on port 80, the default page is returned.



Note

The ping, nmap and curl commands can also be run from the WebUI using the menu option: **Local Configuration > Execute Shell Command**.

- Check if there is a firewall preventing access to the Real Server.

Requests are Not being Load Balanced as Expected

If the system Overview reports that there is an imbalance in the number of connections. This may be caused by a number of reasons as listed below:

Are clients connecting from behind a NAT device - If this is the case, then all requests will appear to come from the same source IP address. This will be an issue if source IP address persistence is used because all client sessions would be load balanced to the same Real Server.

Has a real server been brought back online - If **Balance Mode** has been set to **Weighed Least Connection**, when



the server is brought online it will handle all new requests until balance is reached across all servers. This may occur quickly if lots of new requests are received. If not, the imbalance will continue for longer. If **Balance Mode** has been set to **Weighted Round Robin**, requests will continue to be distributed equally (assuming the weight is the same). If the other Real Servers in the cluster were already processing requests there will again be an imbalance.

Are you testing using a single test client - If persistence is enabled for the VIP then you'll be load balanced to the same Real Server.

Log File - Layer 4 Virtual Services

The layer 4 log contains all log entries from the layer 4 health checking process Ldirectord daemon. This is useful for monitoring when and why real services were down. The log can be viewed using the WebUI menu option: **Logs > Layer 4**. The log can be searched using the search box at the top of the screen or can be downloaded for external analysis. The log can be refreshed using the **Check Status** button. The logging here can be quite verbose but it clearly shows exactly what the health checking process is doing. The log file is located here: /var/log/ldirectord.log.

Interpreting the log file

The following log snippet shows Real Server RIP1 failing its healthcheck and then subsequently passing the check. The latest entries are at the top:

```
2022-07-15T10:57:25.78+00:00 lbmaster ldirectord[18650]: Deleted fallback server (127.0.0.1:80), for virtual service VIP1 (192.168.111.152:80)
2022-07-15T10:57:25.78+00:00 lbmaster ldirectord[18650]: Added real server RIP1 (192.168.110.241:80), for virtual service VIP1 (192.168.111.152:80). Weight set to 100
2022-07-15T10:57:25.77+00:00 lbmaster ldirectord[18650]: Resetting soft failure count: 192.168.110.241:80 (tcp:192.168.111.152:80)
2022-07-15T10:57:12.76+00:00 lbmaster ldirectord[18650]: Deleted real server RIP1 (192.168.110.241:80), for virtual service VIP1 (192.168.111.152:80)
2022-07-15T10:57:12.76+00:00 lbmaster ldirectord[18650]: Added fallback server (127.0.0.1:80), for virtual service VIP1 (192.168.111.152:80). Weight set to 1
2022-07-15T10:57:04.75+00:00 lbmaster ldirectord[18650]: Soft failure real server: 192.168.110.241:80 (tcp:192.168.111.152:80) failure 1/2
```

This shows that:

1. Real Server RIP1 failed its first health check at 10:57:04.
2. By 10:57:12 RIP1 had failed its second health check and since this is the default failure threshold and since this was the failure of the last remaining working Real Server, the fallback server was added and RIP1 was removed.
3. At 10:57:25 RIP1 started passing health checks so was re-added and the fallback server was removed.

Log File - layer 7 Virtual Services

By default, logging for Layer 7 services is disabled.

To enable logging for layer 7 services:

1. Using the WebUI navigate to: **Cluster Configuration > Layer 7 - Advanced Configuration**.
2. Set **Logging** to the required value, the options are:
 - **Off** - No logging enabled.
 - **Service State** - Logs service restarts, reloads and health checks.
 - **Errors** - Logs the service state and Errors.



- **Everything** - Logs errors and all connections. Note that this logs every connection handled by HAProxy, which is likely to rapidly exhaust the log space on a busy Virtual Service.

3. Click **Update**.

4. Reload HAProxy using the Reload button in the "Commit changes" message box at the top of the screen.

Once enabled, the log can be viewed using the WebUI menu option: **Logs > Layer 7**. The log can be searched using the search box at the top of the screen or can be downloaded for external analysis. The log can be refreshed using the **Check Status** button. The log file is located here: `/var/log/haproxy.log`.

Interpreting the log file

The following log snippet shows Real Server RIP1 failing its healthcheck and then subsequently passing the check. The latest entries are at the top:

```
2022-07-15T13:12:38.00+00:00 lbmaster haproxy[30898]: Server VIP2/RIP1 is UP. 1 active and 1 backup servers online. 0 sessions requested, 0 total in queue.
2022-07-15T13:12:30.99+00:00 lbmaster haproxy[30898]: Server VIP2/RIP1 is UP. 1 active and 1 backup servers online. 0 sessions requested, 0 total in queue.
2022-07-15T13:12:30.99+00:00 lbmaster haproxy[30898]: Health check for server VIP2/RIP1 succeeded, reason: Layer4 check passed, check duration: 0ms, status: 2/2 UP.
2022-07-15T13:12:26.99+00:00 lbmaster haproxy[30898]: Health check for server VIP2/RIP1 succeeded, reason: Layer4 check passed, check duration: 0ms, status: 1/2 DOWN.
2022-07-15T13:11:18.96+00:00 lbmaster haproxy[30898]: Server VIP2/RIP1 is DOWN. 0 active and 1 backup servers left. Running on backup. 0 sessions active, 0 requested, 0 remaining in queue.
2022-07-15T13:11:18.96+00:00 lbmaster haproxy[30898]: Health check for server VIP2/RIP1 failed, reason: Layer4 timeout, check duration: 400ms, status: 0/2 DOWN.
2022-07-15T13:11:10.96+00:00 lbmaster haproxy[30898]: Health check for server VIP2/RIP1 failed, reason: Layer4 timeout, check duration: 400ms, status: 1/2 UP.
```

This shows that:

1. Real Server RIP1 failed its first health check at 13:11:10.
2. By 13:11:18 RIP1 had failed its second health check and since this is the default failure threshold and since this was the failure of the last remaining working Real Server, the fallback server was activated and RIP1 was removed.
3. At 13:12:26 RIP1 started working and passed its first health check.
4. At 13:12:30 RIP1 passed its second health check (2 passes before being brought online is the default) so was brought back online and the fallback server was deactivated.

The following log snippet shows a client (192.168.65.194) making three HTTP GET requests to VIP1/RIP1. The latest entries are at the top:

```
2022-07-15T13:31:48.66+00:00 lbmaster haproxy[30898]: 192.168.65.194:53943 [15/Jul/2022:13:31:48.658] VIP2 VIP2/RIP1 0/0/0/1/1 404 1406 - - --NI 2/2/0/0/0 0/0 "GET /favicon.ico HTTP/1.1"
2022-07-15T13:31:48.64+00:00 lbmaster haproxy[30898]: 192.168.65.194:53943 [15/Jul/2022:13:31:48.623] VIP2 VIP2/RIP1 1/0/0/0/18 200 99947 - - --NI 2/2/0/0/0 0/0 "GET /iisstart.png HTTP/1.1"
2022-07-15T13:31:48.60+00:00 lbmaster haproxy[30898]: 192.168.65.194:53943 [15/Jul/2022:13:31:48.605] VIP2 VIP2/RIP1 0/0/0/1/1 200 943 - - --NI 1/1/0/0/0 0/0 "GET / HTTP/1.1"
```

HAProxy HTTP logs are divided into 16 fields. The following table breaks down the first line from the snippet above:

Field	Format	Extract from the example above
1	process_name '[' pid ']:'	haproxy[30898]:
2	client_ip ':' client_port	192.168.65.194:53943
3	'[' request_date ']'	[15/Jul/2022:13:31:48.658]
4	frontend_name	VIP2
5	backend_name '/' server_name	VIP2/RIP1
6	TR '/' Tw '/' Tc '/' Tr '/' Ta*	0/0/0/1/1



Field	Format	Extract from the example above
7	status_code	200
8	bytes_read*	943
9	captured_request_cookie	-
10	captured_response_cookie	-
11	termination_state	--NI
12	actconn '/' feconn '/' beconn '/' srv_conn '/' retries*	2/2/0/0/0
13	srv_queue '/' backend_queue	0/0
14	'{' captured_request_headers* '}'	n/a for this example
15	'{' captured_response_headers* '}'	n/a for this example
16	"" http_request ""	"GET /favicon.ico HTTP/1.1"

Note

For a detailed description of each field in a log entry for an HTTP request, please refer to [HTTP Log Format](#).

Note

TCP logs are similar but there are only 10 fields. For more information, please refer to [TCP Log Format](#).

Note

For full details of all HAProxy logging features, please refer to [HAProxy Logging](#).

SSL Termination (Pound)

By default, Pound logging is disabled.

To enable logging for Pound:

1. Using the WebUI navigate to: **Cluster Configuration > SSL - Advanced Configuration**.
2. Under the **Pound Global Settings** section set **Logging** to **Yes**.
3. Click **Update**.

Once enabled, the log can be viewed using the WebUI menu option: **Logs > SSL Termination (Pound)**. The log can be searched using the search box at the top of the screen or can be downloaded for external analysis. The log can be refreshed using the **Check Status** button. The log file is located here: `/var/log/pound.log`.

SSL Termination (STunnel)

By default, STunnel logging is set to **Emergency(0)**.

To change the logging level for STunnel:

1. Using the WebUI navigate to: **Cluster Configuration > SSL - Advanced Configuration**.



2. Under the **STunnel Global Settings** section, set the **Debug Level** to the required value, the options are:

Value	Severity	Description
0	Emergency	System is unusable
1	Alert	Action must be taken immediately
2	Critical	Critical conditions
3	Error	Error conditions
4	Warning	Warning conditions
5	Notice	Normal but significant conditions
6	Informational	Informational messages
7	Debug	Debug-level messages

**Note**

Log Entries for the selected level plus all the levels below will be included. For example, if level 4 is selected, levels 0, 1, 2 and 3 will also be logged.

3. Click **Update**.

Once enabled, the log can be viewed using the WebUI menu option: **Logs > SSL Termination (STunnel)**. The log can be searched using the search box at the top of the screen or can be downloaded for external analysis. The log can be refreshed using the **Check Status** button. The log file is located here: `/var/log/stunnel.log`.

WAF

For information on accessing and interpreting WAF logs, please refer to [WAF Gateway Error Logs](#).

Heartbeat

For information on diagnosing Heartbeat issues and accessing the Heartbeat log, please refer to [Clustered Pair Diagnostics](#).

Shuttle

The Shuttle service is used to pass appliance details supplied by the Gateway service to the Portal. The log shows all communications that have occurred between the Shuttle service and the Portal.

Chapter 12 - Monitoring & Reporting

SNMP Reporting

By default, SNMP is disabled on the appliance. For information on enabling SNMP and configuring basic SNMP settings, please refer to [SNMP Configuration](#).

MIB Files

The appliance has a number of Management Information Base files (MIBs) available for use with your monitoring system. The appliance includes all the standard Linux MIBs and also a number of custom MIB that enable layer 4 and layer 7 services to be monitored. All MIB files are located in `/usr/share/snmp/mibs` on the appliance. The custom MIBs are also available for download using the links shown below.

General - For network stats, memory usage, CPU load etc, standard Linux MIBs can be used.

Layer 4 Services - For monitoring layer 4 services, [OC-MIB.txt](#) & [LVS-MIB.txt](#) are provided.

Layer 7 Services - for monitoring layer 7 services, [LBO-MIB.txt](#), [L7STAT-EXPERIMENTAL.txt](#) & [L7INFO-EXPERIMENTAL-MIB.txt](#) are provided.

Note

`/etc/snmp/snmp.conf` includes the directive **mibs +ALL** which means that all available MIBs are loaded.

The following two sections include various examples to help demonstrate how SNMP values for Layer 4 and Layer 7 services can be retrieved at the console or via an SSH session.

SNMP for Layer 4 Services

The root OID for Layer 4 services is: **1.3.6.1.4.1.8225.4711**.

Note

Lots of information is available, the layer 4 MIB file includes comprehensive descriptions of all OIDs.

View information for layer 4 services starting at the root OID:

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost 1.3.6.1.4.1.8225.4711

LVS-MIB::lvsVersion.0 = STRING: "1.2.1"
LVS-MIB::lvsNumServices.0 = INTEGER: 0
LVS-MIB::lvsHashTableSize.0 = INTEGER: 32768
...
etc.
```

The same can be achieved using the MIB object name `lvs`:

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost lvs
```

Monitoring Layer 4 VIPs & RIPs using SNMP

Accessing VIP Information

List all VIPs using the OID for layer 4 VIPs:

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost 1.3.6.1.4.1.8225.4711.17.1.4

LVS-MIB::lvsServiceAddr.1 = IPAddress: 192.168.111.223
LVS-MIB::lvsServiceAddr.2 = IPAddress: 192.168.111.222
```

The same can be achieved using the MIB object name *lvsServiceAddr*:

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost lvsServiceAddr
```

Display the lvsService table:

```
[root@lbmaster ~]# snmptable -c public -v 2c localhost lvsService
```

lvsServiceNum...	lvsServiceSch...	lvsServiceProto	lvsServiceAddr	lvsServicePort	lvsServiceFW...	lvsServicePers...	lvsSe
1	wlc	tcp	192.168.111.223	3389	undefined	300	255.1
2	wlc	tcp	192.168.111.222	80	undefined	300	255.1

Accessing RIP Information

List the Real Servers that are passing their health check using the OID for layer 4 RIPs:

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost 1.3.6.1.4.1.8225.4711.18.1.3

LVS-MIB::lvsRealServerAddr.1.1 = IPAddress: 192.168.110.240
LVS-MIB::lvsRealServerAddr.1.2 = IPAddress: 192.168.110.243
LVS-MIB::lvsRealServerAddr.2.1 = IPAddress: 192.168.110.243
LVS-MIB::lvsRealServerAddr.2.2 = IPAddress: 192.168.110.240
```

Note

For layer 4 services, if the health check fails, the failed server will be omitted from the list.

The same can be achieved using the MIB object name *lvsRealServerAddr*:

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost lvsRealServerAddr
```

Display the lvsReal table:

```
[root@lbmaster ~]# snmptable -c public -v 2c localhost lvsReal
```

lvsRealService...	lvsRealServer...	lvsRealServer...	lvsRealServer...	lvsRealServer...	lvsRealServer...	lvsRealStats...
1	1	192.168.110.240	3389	route	100	0
1	2	192.168.110.243	3389	route	100	0
2	1	192.168.110.243	80	route	100	0
2	2	192.168.110.240	80	route	100	0

SNMP for Layer 7 Services

The root OID for Layer 7 services is: **1.3.6.1.4.1.54849**.

Note

Lots of information is available, the layer 7 MIB files include comprehensive descriptions of all OIDs.

View information for layer 7 services starting at the root OID:

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost 1.3.6.1.4.1.54849

L7INFO-EXPERIMENTAL-MIB::l7InfoLastPolled.0 = Timeticks: (141790592) 16 days, 9:51:45.92
L7INFO-EXPERIMENTAL-MIB::l7InfoUpdateInterval.0 = INTEGER: 4000
L7INFO-EXPERIMENTAL-MIB::l7InfoLastError.0 = Gauge32: 0
L7INFO-EXPERIMENTAL-MIB::l7InfoName.0 = STRING: HAProxy
L7INFO-EXPERIMENTAL-MIB::l7InfoVersion.0 = STRING: 1.8.25
...
etc.
```

The same can be achieved using the MIB object name:

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost loadbalancerOrg
```

Display layer 7 HAProxy process information:

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost L7info
```

Note

This gives a similar output to the familiar HAProxy "show info" command. for more information on using the Linux socat command to run the "show info" command, please refer to [Using Linux socket commands to configure Layer 7 Services](#).

Monitoring Layer 7 VIPs & RIPs using SNMP

Accessing VIP Information

List all VIPs using the OID for layer 7 VIPs:

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost 1.3.6.1.4.1.54849.1.2.5.1.2

L7STAT-EXPERIMENTAL-MIB::l7FePxname.4 = STRING: stats
```

```
L7STAT-EXPERIMENTAL-MIB::l7FePxdname.493413195 = STRING: VIP4
L7STAT-EXPERIMENTAL-MIB::l7FePxdname.1593140907 = STRING: VIP3
```

The same can be achieved using the MIB object name 'l7FePxdname':

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost l7FePxdname
```

Display the L7Frontend table:

```
[root@lbmaster ~]# snmptable -c public -v 2c localhost l7Frontend
```

l7FeId	l7FePxdname	l7FeStatus	l7FeMode	l7FeConnRate	l7FeConnRate...	l7FeConn
4	stats	no lb	http	0	2	3
493413195	VIP4	no lb	tcp	0	0	0
1593140907	VIP3	no lb	http	0	0	0

Display the L7Backend table:

```
[root@lbmaster ~]# snmptable -c public -v 2c localhost l7Backend
```

l7BeId	l7BePxdname	l7BeStatus	l7BeWeight	l7BeMode	l7BeCookie	l7BeAlgo
4	stats	up	0	http		roundRobin
493413195	VIP4	up	100	tcp		leastConnection
1593140907	VIP3	up	100	http	SERVERID	leastConnection

Accessing RIP Information

List all RIPs using the OID for layer 7 RIPs:

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost 1.3.6.1.4.1.54849.1.2.7.1.4
```

```
L7STAT-EXPERIMENTAL-MIB::l7SrvSvname.493413195.1 = STRING: backup
L7STAT-EXPERIMENTAL-MIB::l7SrvSvname.493413195.551991902 = STRING: RDP2
L7STAT-EXPERIMENTAL-MIB::l7SrvSvname.493413195.1899219725 = STRING: RDP1
L7STAT-EXPERIMENTAL-MIB::l7SrvSvname.1593140907.1 = STRING: backup
L7STAT-EXPERIMENTAL-MIB::l7SrvSvname.1593140907.1269887581 = STRING: Web2
L7STAT-EXPERIMENTAL-MIB::l7SrvSvname.1593140907.1749728569 = STRING: Web1
```

The same can be achieved using the MIB object name 'l7SrvSvname':

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost l7SrvSvname
```

Display the L7Server table:



```
[root@lbmaster ~]# snmptable -c public -v 2c localhost L7Server
```

l7SrvIid	l7SrvSid	l7SrvPxname	l7SrvSvname	l7SrvAddressT...	l7SrvAddress...	l7SrvAddress
493413195	1	VIP4	backup	ipv4	127.0.0.1	9081
493413195	551991902	VIP4	RDP2	ipv4	-64.-88.110.-13	3389
493413195	1899219725	VIP4	RDP1	ipv4	-64.-88.110.-16	3389
1593140907	1	VIP3	backup	ipv4	127.0.0.1	9081
1593140907	1269887581	VIP3	Web2	ipv4	-64.-88.110.-13	80
1593140907	1749728569	VIP3	Web1	ipv4	-64.-88.110.-16	80

SNMPv3

For SNMPv3, the SNMP walk command format is:

```
snmpwalk -v3 -u <USERNAME> -l authNoPriv -a MD5 -A <PASS-PHRASE> -m LVS-MIB localhost  
1.3.6.1.4.1.8225.4711
```

e.g.

```
snmpwalk -v3 -u snmpv3user -l authNoPriv -a MD5 -A sNmPv31864zX -m LVS-MIB localhost  
1.3.6.1.4.1.8225.4711
```

Reports

All reports can be accessed using the **Reports** option in the WebUI.

Layer 4 Status

This report shows the current weight and number of active & inactive connections for each Real Server. If a Real Server has failed a health check, it will not be listed. Use the **Logs > Layer 4** option to view the Ldirectord log file if expected servers are missing.

Layer 4 Status

Check Status

Virtual Service	Real Server	Forwarding Method	Weight	Active Connections	Inactive Connections
HTTP-Cluster1 192.168.110.120 port 192.168.110.120/tcp					
	RIP1 192.168.110.240	Route	100	0	0
	RIP2 192.168.110.241	Route	100	0	0
	RIP3 192.168.110.242				

IP Virtual Server version 1.2.1 (size=32768)

Prot LocalAddress:Port Scheduler Flags

-> RemoteAddress:Port Forward Weight ActiveConn InActConn

TCP 192.168.110.120:80 wlc persistent 300

-> 192.168.110.240:80 Route 100 0 0

-> 192.168.110.241:80 Route 100 0 0

In the example above, the details for RIP3 are not displayed because it's failing its health checks.

The **Check Status** button can be used to refresh the display.

Layer 4 Traffic Rate

This report shows the current connections per second and bytes per second to each Real Server. If a Real Server has failed a health check, it will not be listed.



Check Status

Virtual Service	Real Server	Connections / s	Incoming Packets / s	Outgoing Packets / s	Incoming Bytes / s	Outgoing Bytes / s
HTTP-Cluster1 192.168.110.120 port 192.168.110.120/tcp		0	0	0	0	0
	RIP1 192.168.110.240	0	0	0	0	0
	RIP2 192.168.110.241	0	0	0	0	0
	RIP3 192.168.110.242					

IP Virtual Server version 1.2.1 (size=32768)

Prot	LocalAddress:Port	CPS	InPPS	OutPPS	InBPS	OutBPS
-> RemoteAddress:Port						
TCP	192.168.110.120:80	0	0	0	0	0
	-> 192.168.110.240:80	0	0	0	0	0
	-> 192.168.110.241:80	0	0	0	0	0

In the example above, the details for RIP3 are not displayed because it's failing its health checks.

The **Check Status** button can be used to refresh the display.

Layer 4 traffic Counters

This report provides details of the traffic being handled by each Real Server since the counters were last re-set. If a Real Server has failed a health check, it will not be listed.

[Check Status](#)[Reset Counters](#)

Virtual Service	Real Server	Connections	Incoming Packets	Outgoing Packets	Incoming Bytes	Outgoing Bytes
HTTP-Cluster1 192.168.110.120 port 192.168.110.120/tcp		0	0	0	0	0
	RIP1 192.168.110.240	0	0	0	0	0
	RIP2 192.168.110.241	0	0	0	0	0
	RIP3 192.168.110.242					

IP Virtual Server version 1.2.1 (size=32768)

Prot	LocalAddress:Port	Conns	InPkts	OutPkts	InBytes	OutBytes
-> RemoteAddress:Port						
TCP	192.168.110.120:80	0	0	0	0	0
	-> 192.168.110.240:80	0	0	0	0	0
	-> 192.168.110.241:80	0	0	0	0	0

Note

For DR mode, since return traffic bypasses the load balancer, there will be no stats for return packets.

In the example above, the details for RIP3 are not displayed because it's failing its health checks.

The **Check Status** button can be used to refresh the display.

Layer 4 Current Connections

The current connections report is very useful for diagnosing issues with routing or ARP related problems. In the example below, the state is shown as **SYN_RECV**. For layer 4 DR mode this is normally a good indication that the "ARP Problem" has not been solved. For NAT mode, this is a good indication that the Real Server's default gateway has not been configured to be the load balancer and therefore return traffic is not routed correctly.

Check Status

IPVS connection entries

pro	expire	state	source	virtual	destination
TCP	04:44	NONE	192.168.64.7:0	192.168.110.120:80	192.168.110.241:80
TCP	00:49	SYN_RECV	192.168.64.7:28808	192.168.110.120:80	192.168.110.241:80
TCP	00:49	SYN_RECV	192.168.64.7:28809	192.168.110.120:80	192.168.110.241:80

Note

The IPVS connection entries in state **NONE** represent the persistence related entries for client connections, and are not actual client connections. These only appear when persistence is enabled.

The **Check Status** button can be used to refresh the display.

Layer 4 Current Connections (Resolve Hostnames)

This is the same as the current connections report but is slower as it attempts to lookup the DNS name of each IP address.

Layer 7 Status

This report contains the current status of all configured layer 7 HAProxy Virtual Services and Real Servers.

HAProxy

Statistics Report for pid 7370

> General process information

pid = 7370 (process #1, nbproc = 1, nbthread = 1)
uptime = 0d 0h00m34s
system limits: memmax = unlimited; ulimit-n = 80036
maxsock = 80036; maxconn = 40000; maxpipes = 0
current conns = 1; current pipes = 0/0; conn rate = 1/sec; bit rate = 0.000 kbps
Running tasks: 1/16; idle = 100 %

active UP
active UP, going down
active DOWN, going up
active or backup DOWN
active or backup DOWN for maintenance (MAINT)
backup UP
backup UP, going down
backup DOWN, going up
not checked

Display option:

- Scope :
- Hide 'DOWN' servers
- Disable refresh
- Refresh now
- CSV export

External resources:

- Primary site
- Updates (v2.0)
- Online manual

Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

L7-HTTP

	Queue			Session rate			Sessions				Bytes		Denied		Errors		Warnings		Server												
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle	
Frontend				0	0	-	0	0		40 000	0			0	0	0	0	0	0	0	0	0	OPEN								
backup	0	0	-	0	0		0	0		-	0	0	?	0	0	0	0	0	0	0	0	0	no check		1	-	Y				-
rip1	0	0	-	0	0		0	0		-	0	0	?	0	0	0	0	0	0	0	0	0	34s UP	L4OK in 0ms	100	Y	-	0	0	0s	-
Backend	0	0		0	0		0	0		4 000	0	0	?	0	0	0	0	0	0	0	0	0	34s UP		100	1	1		0	0s	

:prometheus-exporter

	Queue			Session rate			Sessions				Bytes		Denied		Errors		Warnings		Server												
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle	
Frontend				0	0	-	0	0		40 000	0			0	0	0	0	0	0	0	0	0	OPEN								
Backend	0	0		0	0		0	0		4 000	0	0	?	0	0	0	0	0	0	0	0	0	34s UP		0	0	0		0	0	

:stats

	Queue			Session rate			Sessions				Bytes		Denied		Errors		Warnings		Server												
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle	
Frontend				0	0	-	0	0		40 000	0			0	0	0	0	0	0	0	0	0	OPEN								
Backend	0	0		0	0		0	0		4 000	0	0	?	0	0	0	0	0	0	0	0	0	34s UP		0	0	0		0	0	

:admin_stats

	Queue			Session rate			Sessions				Bytes		Denied		Errors		Warnings		Server												
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle	
Frontend				1	1	-	1	1		40 000	2			974	29 143	0	0	0	0	0	0	0	OPEN								
Backend	0	0		0	0		0	0		4 000	0	0	0s	974	29 143	0	0	0	0	0	0	0	34s UP		0	0	0		0	0	

Note

From v8.8.1, all statistics are automatically refreshed every 30 seconds by default.

Layer 7 Stick Table

Displays the layer 7 stick tables. Stick tables are used by various persistence types to track client / Real Server connections. All VIPs that use stick tables are listed in the dropdown.



Web-Cluster
?
Refresh
Clear Table

1 Entries Returned

ID	Key	Use	Expires	Server	Remove
0x247cd60	10.0.51.27	use=0	1787059	Web1	✖

Page 1 of 1

Prev
Next

Individual stick table entries can be removed by clicking the red "X" in the remove column, the whole table can be cleared by clicking the **Clear Table** button.

GSLB Reports

For details, please refer to [GSLB Diagnostics](#).

Graphs

All graphs can be accessed using the *Reports > Graphing* option in the WebUI.

Resource Utilization Graphs

Network Throughput - The following graphs are provided:

- Total packets/s (hourly, daily, weekly, monthly and yearly)
- Total bytes/s (hourly, daily, weekly, monthly and yearly)

System Load Average - The following graphs are provided:

- System Load Average (hourly, daily, weekly, monthly and yearly)

The system load average is a very useful way to determine if the appliance is overloaded. To understand how to interpret the value, a bridge analogy can be used:

- A value of 0 means there's no traffic on the bridge at all. In fact, between 0 and 1 means there's no congestion, and an arriving car will just drive straight onto the bridge.
- A value of 1 means the bridge is exactly at capacity. All is still good, but if traffic gets a little heavier, things are going to build up and getting accross the bridge will slow down.
- A value over 1 means there's congestion: A value of 2 means that there are two lanes of cars - 1 lane on the bridge and 1 lane waiting. A value of 3 means there are 3 lanes of cars - 1 lane on the bridge and two lanes waiting, etc.

Using a single core with the load average somewhere around 0.75 - 1 is great. Using a quad core with the load average somewhere around 3.0 - 4.0 would also be fine.



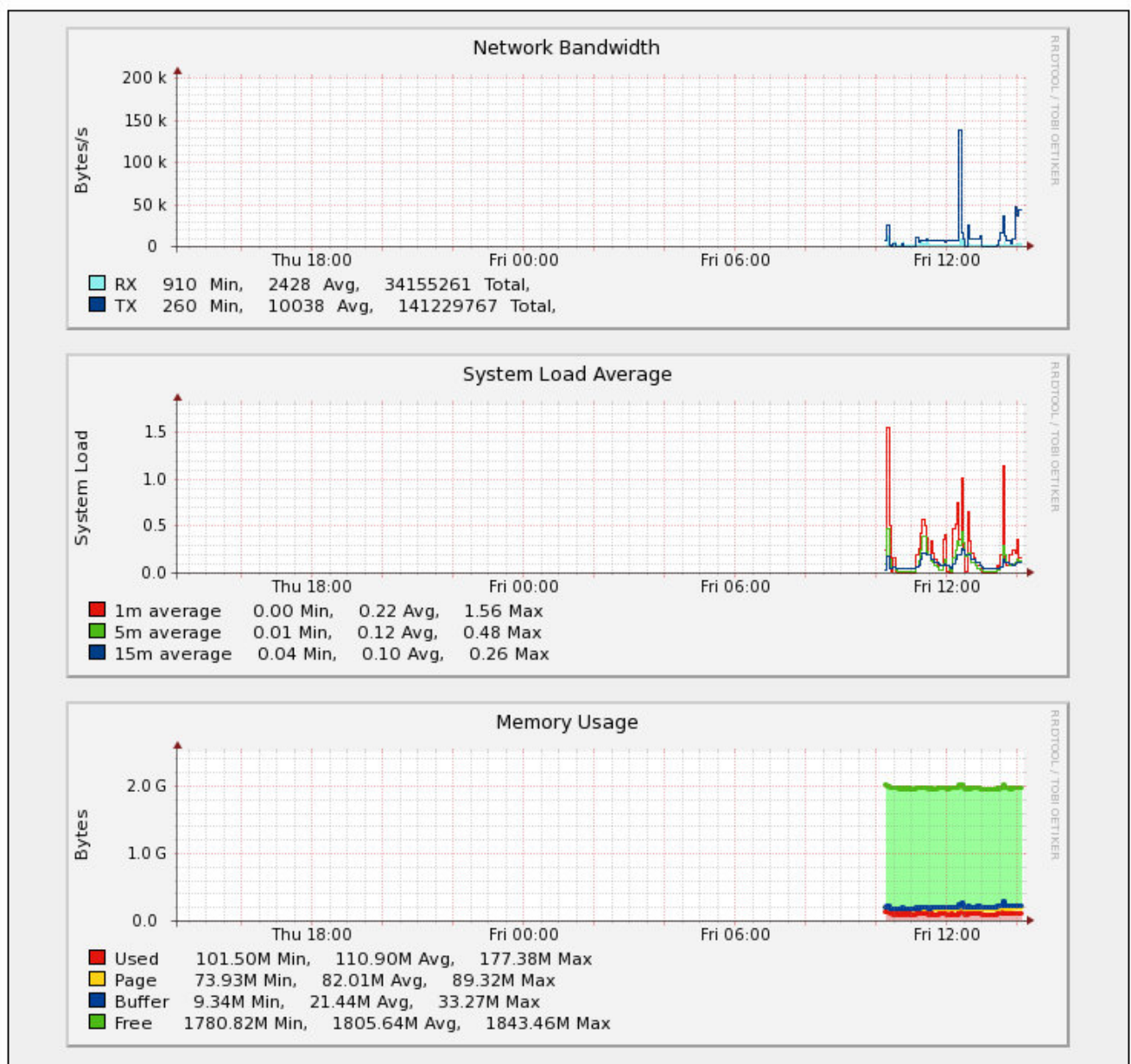
Memory Usage - The following graphs are provided:

- Memory Usage (hourly, daily, weekly, monthly and yearly)

Disk Usage - The following graphs are provided:

- Disk Usage (hourly, daily, weekly, monthly and yearly)

By default, daily graphs for Network Throughput (bytes/s), System Load Average and Memory Usage are displayed in the System Overview. To drill down and view all available graphs, click on the particular graph in question.



Note

These graphs can be disabled/hidden if preferred using the WebUI menu option: *Local Configuration > Graphing*.

Note

The disk usage graph is not displayed in the System Overview, it can be accessed using the WebUI option: **Reports > Graphing** and selecting **Disk Usage** in the dropdown.

Virtual Service & Real Server Graphs

The following graphs are provided for each Virtual Service and each Real Server:




- Connections (hourly, daily, weekly, monthly and yearly)
- Bytes/s (hourly, daily, weekly, monthly and yearly)

Graphs are automatically created when new Virtual Services / Real Servers are configured.

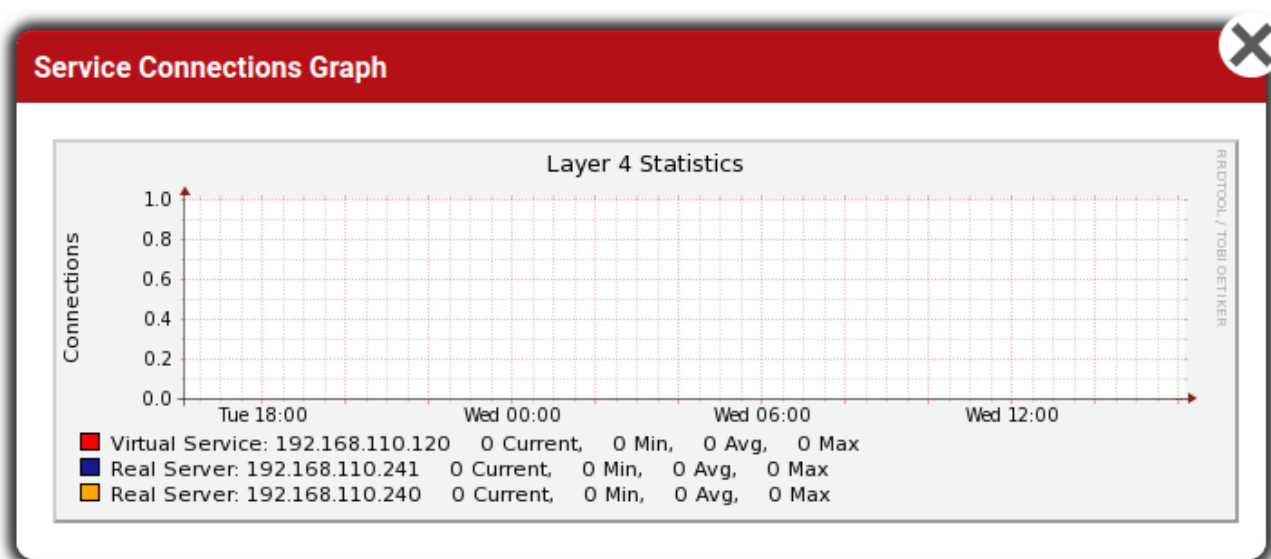
Graphs for the configured Virtual Services & Real Servers can be accessed either from the System Overview using the appropriate graph icon that appears on the right side of each VIP and RIP or from the dropdown available in the WebUI under **Reports > Graphing**.

Using the System Overview

The graph is displayed by clicking the relevant graph icon that's displayed next to each VIP/RIP:

↑	HTTP-Cluster1	192.168.110.120	80	0	TCP	Layer 4	DR	
	REAL SERVER	IP	PORTS	WEIGHT	CONNS			
↑	RIP1	192.168.110.240	80	100	0	Drain	Halt	
↑	RIP2	192.168.110.241	80	100	0	Drain	Halt	

When this method is used, the daily Service Connections Graph is displayed:

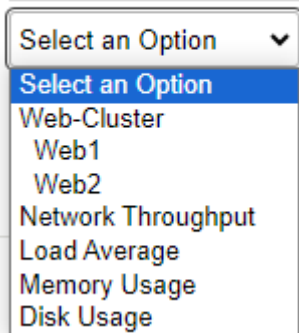


Clicking anywhere within this graph opens the complete list of graphs for the VIP/RIP in question. This is the same as selecting the VIP/RIP in the **Reports > Graphing** menu options as explained below.

Using the WebUI menu option: *Reports > Graphing*

When selected, a dropdown similar to the following is displayed:

Graphing



The dropdown list is kept in-sync with the available VIPs & RIPs.

Note

For more information on using the System Overview, please refer to [Chapter 8 - Real Server Health Monitoring & Control](#).

When this method is used, a complete list of graphs is displayed for the Virtual Service or Real Server that is selected.

Graphing Options

The graphing sub-system can be customized to suit your requirements.

To change the graphing options:

1. Using the WebUI, navigate to: *Local Configuration > Graphing*.

Layer 4	On ▼	?
Layer 7	On ▼	?
Interfaces	On ▼	?
Load Average	On ▼	?
Memory	On ▼	?
Disk Usage	On ▼	?

Advanced Configuration

Interval	10	?
Timeout	2	?
Threads	6	?
Logging	Off ▼	?

Update

- Set the required option for each graphing category - either enabled (default), disabled or deleted.
- Set the data collector **Interval** for the data collector in seconds.

Warning

This is a global value and will effect all collectors. Do not change this value unless advised to do so by Support. Changing this value will reset the graphing databases and all the current data will be lost.

- Set the data collector **Timeout** for the data collector in seconds.

Note

Do not change unless advised to do so by support.

- Set the number of **Threads** for the data collector process.

Note

Do not change unless advised to do so by support.

- Configure data collector logging for collectd.

Note

This is incredibly verbose and should only be enabled for debugging purposes.

Layer 7 (HAProxy) Prometheus Exporter

The Prometheus Exporter consumes the HAProxy stats page and converts the data to the [Prometheus time series](#).

To enable the Prometheus exporter:

1. Using the WebUI navigate to: **Cluster Configuration > Layer 7 - Advanced Configuration**.
2. Scroll to the bottom of the page and enable (check) the **Enable Prometheus Exporter** checkbox.
3. Reload HAProxy using the button in the "Commit changes" message box at the top of the screen.

Once enabled, the Prometheus endpoint will be available here:

`https://<appliance-ip-address>:9443/lbadmin/stats/l7prometheus/`

 **Note**

If the WebUI has been configured to listen on a different port, modify the URL accordingly. For more details on setting the port, please refer to [Service Socket Addresses](#).

You can authenticate using the credentials of any valid WebUI user account.

 **Note**

For more information, please refer to our [Prometheus Blog](#).

Grafana

Grafana is a tool that allows you to visualise data from sources such as Prometheus. For information on installing and configuring Grafana and setting up a Prometheus data source on the appliance, please refer to our [Grafana Blog](#).

Chapter 13 - Useful Tools & Utilities

Useful Diagnostics Tools

Full root access to the appliance is supported which enables many useful commands to be run directly at the console or via an SSH session. Many commands can also be run using the WebUI menu option: **Local Configuration > Execute Shell Command**. Several commonly used examples are listed below.

Note

The "Execute Shell Command" menu option is disabled by default. This can be enabled using the WebUI option: **Local Configuration > Security**. Set **Appliance Security Mode** to **Custom** then click **Update**.

Note

"root" user console and SSH password access are disabled by default. These options can be enabled using the WebUI option: **Local Configuration > Security**. You'll need to Set **Appliance Security Mode** to **Custom**, enable the required option(s) and click **Update**.

Netstat

Print network connections, routing tables, interface statistics, masquerade connections, and multicast memberships. Useful to check that services are listening on the correct IP/port.

e.g. `netstat -anp`

Command Output:

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:7778          0.0.0.0:*               LISTEN      7218/haproxy
tcp        0      0 192.168.100.238:80      0.0.0.0:*               LISTEN      7218/haproxy
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      9638/sshd
tcp        0      0 0.0.0.0:9081            0.0.0.0:*               LISTEN      5136/nginx
tcp        0      0 192.168.100.237:22      10.10.0.26:61430        ESTABLISHED 9801/sshd
tcp        0      0 :::9443                  :::*                    LISTEN      732/httpd
tcp        0      0 :::22                    :::*                    LISTEN      9638/sshd
tcp        0      0 :::9080                  :::*                    LISTEN      732/httpd
udp        0      0 192.168.100.238:123     0.0.0.0:*               1650/ntpd
udp        0      0 192.168.100.237:123     0.0.0.0:*               1650/ntpd
udp        0      0 127.0.0.1:123           0.0.0.0:*               1650/ntpd
udp        0      0 0.0.0.0:123             0.0.0.0:*               1650/ntpd
udp        0      0 0.0.0.0:161             0.0.0.0:*               10089/snmpd
etc.
```

Telnet

The telnet command is used to communicate with another host using the TELNET protocol. It's very useful for testing that a connection to a specific port can be made. Note that this command should be run from the console or a terminal session rather than via the WebUI.



```
e.g. telnet 192.168.100.10 80
```

In this example, 192.168.100.10 is a Real Server, the command is useful to ensure that the load balancer is able to successfully connect to this server on port 80.

```
[root@lbmaster ~]# telnet 192.168.100.10 80
Trying 192.168.100.10...
Connected to 192.168.100.10.
Escape character is '^]'.
```

Tcpdump

Tcpdump enables network traffic to be dumped to a file for analysis. Filters can also be applied if required to select which traffic is captured. Very useful tool when diagnosing network issues. Note that this command should be run from the console or a terminal session rather than via the WebUI.

```
e.g. tcpdump -i any -s 0 -w tcpdump-file.pcap
```

This command captures all network traffic on all interfaces using the maximum packet size of 65535 bytes and dumps it to a file called tcpdump-file.pcap. To end the capture use CTRL+C.

Our support department may ask you to run this command and send the resulting output file to help them diagnose certain network issues.

Ethtool

Ethtool is used for querying settings of an Ethernet device and changing them.

```
e.g. ethtool eth0
```

Command output:

```
Settings for eth0:
    Supported ports: [ TP ]
    Supported link modes:   1000baseT/Full
                           10000baseT/Full
    Supports auto-negotiation: No
    Advertised link modes:  Not reported
    Advertised pause frame use: No
    Advertised auto-negotiation: No
    Speed: 10000Mb/s
    Duplex: Full
    Port: Twisted Pair
    PHYAD: 0
    Transceiver: internal
    Auto-negotiation: off
    MDI-X: Unknown
    Supports Wake-on: uag
    Wake-on: d
```



Link detected: yes

Nmap

Nmap (Network Mapper) can be used to scan a range of hosts or a single host to determine which ports are open and which services are listening on those ports.

e.g. `nmap 192.168.110.241`

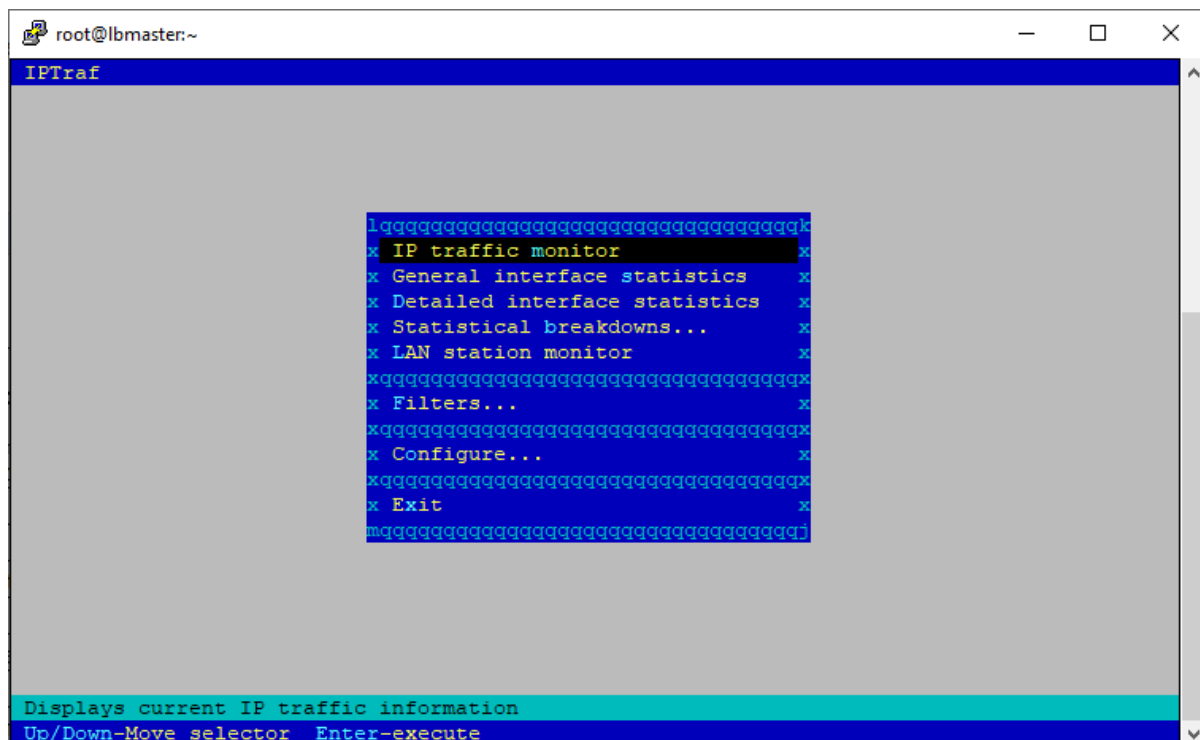
Command output:

```
Starting Nmap 5.51 ( http://nmap.org ) at 2022-03-15 14:54 UTC
Nmap scan report for 192.168.110.241
Host is up (0.0010s latency).
Not shown: 996 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
443/tcp   open  https
3389/tcp  open  ms-term-serv
MAC Address: 00:50:56:82:0B:D3 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 4.53 seconds
```

Iptraf

lpraf (Interactive Colorful IP LAN Monitor) is an ncurses-based IP LAN monitor that generates various network statistics including TCP info, UDP counts, ICMP and OSPF information, Ethernet load info, node stats, IP checksum errors, and others. If the command is issued without any command-line options, the program comes up in interactive mode, with the various facilities accessed through the main menu as shown below:



Wireshark

Wireshark is an open source application that can be used to analyze tcpdump output files. It can be downloaded from [here](#).

Windows Specific Tools

Microsoft Network Monitor

Network Monitor is a simpler alternative to Wireshark that has some nice features. It can be downloaded from [here](#).

WinSCP

WinSCP is an open source application that allows files to be uploaded/downloaded to/from the load balancer using Windows. It can be downloaded from [here](#).

PuTTY

PuTTY is an open source SSH client for Windows. It can be downloaded from [here](#).



Chapter 14 - Backup & Restore and Disaster Recovery

Backup & Restore

Backup

When a backup is performed, a zipped archive is created that includes the following:

- XML Configuration File
- SSL Certificates
- Manual Configurations
- SSH Keys
- Firewall Scripts
- PBR Rules
- Authentication Settings
- LBCLI Password File
- Fallback Page
- User Names and Passwords
- License Key

To perform a backup:

1. Using the WebUI navigate to: **Maintenance > Backup & Restore**.
2. Under the **Backup** tab, enter a password to encrypt sensitive files.



Note

SSL certificates and SSH keys are only backed up if a password is set. If the password is not set, the backup will exclude these files.

3. Click **Backup**.
4. The zip file will be created and downloaded to your browser's default location or you'll be prompted to select a location depending on the settings configured.

Restore

The restore option allows the following filetypes to be restored:

- Backup archives (*.zip) created using the backup feature
- Certificate archives (*.tar.gz) created using v8.6.3 and earlier (included for backward compatibility)
- XML backup (*.xml) created using v8.6.3 and earlier (included for backward compatibility)

To perform a restore:



1. Using the WebUI navigate to: **Maintenance > Backup & Restore**.
2. Select the **Restore** Tab.
3. Click **Choose File** and browse to and select the relevant backup file.
4. If you want the floating IPs not to be brought up when the file is restored, enable (check) the **Disable Floating IPs on restore** checkbox.
5. If the file being restored is a backup archive (zip file), enter the password (if applicable).

 **Note**

If a password was entered when the backup was created, but the password is omitted when performing the restore, you can continue with the restore but sensitive files (certificates and SSH keys) will be omitted.

 **Note**

If a password was not entered when the backup was created, you'll be notified that the backup does not contain the sensitive files and asked if you'd like continue.

6. Click **Upload & Restore**.
7. Click **OK** to confirm that you'd like to continue. The restore will now start.

Backup & Restore

Installing and upgrading XML...

Restoring Configuration from uploaded file...

Resetting Configuration.

Disabling Peer Comms During Restore.

Upgrading XML.

Restoring Network Configuration.

Once complete, heartbeat and possibly other services (depending on the configuration) will need to be either restarted or reloaded. You'll be prompted accordingly in the "Commit changes" message box at the top of the screen. If the box does not appear, click on System Overview or any other menu option - the box will then be displayed. Don't use F5 as this will cause the restore to run again.

Restore System Defaults

This option can be used to reset all settings to their factory defaults.

To restore the appliance to factory defaults:

1. Using the WebUI navigate to: **Maintenance > Backup & Restore**.



2. In the **Restore to system defaults** section under the **Backup** tab, click **Restore**.
3. Click **OK** to confirm you'd like to continue.

Checkpoints

Checkpoints are very similar to [backups](#) and include the same items. In addition, each checkpoint also includes a reference to which software release was installed on the appliance when the checkpoint was created. If the appliance's software is updated, all new checkpoints created include a reference to the updated release. If the appliance is reverted to a checkpoint that refers to a release that is different to the current release, the checkpoint sub system automatically reverts the appliance's software to that release. This enables rollback and roll forward between appliance software versions.

To access Checkpoints:

1. Using the WebUI, navigate to: **Maintenance > Backup & Restore**.
2. Select the **Checkpoints** tab.

Backup & Restore

Backup & Restore						
Backup		Restore	Checkpoints	Template Deployment	Synchronisation	
				Create	Lock	Unlock
				Revert	Delete	
	Release	Name	Date	Size	Status	
<input type="checkbox"/>	8.11.1-1.lb			275358.80 KiB	Current	
<input type="checkbox"/>		elm3T4	2024-03-20 09:55 UTC	73.65 KiB		
<input type="checkbox"/>		YJd9wg	2024-03-20 09:55 UTC	73.65 KiB		
				Create	Lock	Unlock
				Revert	Delete	

In the above example, **Release-8.11.1-1.lb** is the software release that's currently installed on the appliance. This is indicated under the **Status** column. Two checkpoints have been created that reference this release. Each time the appliance's software is updated, an additional release entry is added and becomes the current release.

To create a Checkpoint:

1. Click **Create**, a random 6 character name will be assigned to the new checkpoint and the date, time and size will be logged.



Note

For a clustered pair, create corresponding checkpoints on both appliances.

To lock a Checkpoint:

1. Select the checkpoint(s) to lock using the appropriate checkbox(es) and click **Lock**. Once locked, the checkpoint(s) cannot be deleted.

To revert to a Checkpoint:

1. Select the checkpoint to revert to and click **Revert**.

Note

For a clustered pair, if the checkpoint being reverted does not change the software version, the "Synchronise Configuration with Peer" feature can be used on the Primary appliance after the checkpoint has been reverted to ensure that the Secondary is synchronised. This can be accessed using the WebUI menu option: **Maintenance > Backup & Restore** and selecting the **Synchronisation** tab. If the checkpoint changes the software version, you'll also need to revert to a corresponding checkpoint on the Secondary appliance to ensure that the appliances are synchronised.

Template Deployment

This feature enables Virtual Services, Real Servers, HAProxy SSL terminations, GSLB configuration settings and health check scripts to be exported/imported to/from a JSON configuration file. When importing, new IP addresses can be assigned to each Virtual Service and multiple associated Backend (Real) Servers can be configured.

To access template deployment:

1. Using the WebUI, navigate to: **Maintenance > Backup & Restore**.
2. Select the **Template Deployment** tab.

Backup & Restore

Backup

Restore

Checkpoints

Template Deployment

Synchronisation

Upload a deployment template (*.json):

Choose File

No file chosen

Upload Deployment File

Please select the features you wish to export (prerequisites will be included automatically):

Layer 4

WebCluster

Layer 7

RDP

☐ Do not write default values for exported Floating IPs

Export Selected

Select All

To generate a template deployment file:

1. Select the required features to include in the deployment file. Use the CTRL key to select multiple items or use the **Select All** button.
2. If you don't want to include the IP address for each Virtual Service, enable the *Do not write default values for exported Floating IPs* checkbox.
3. Click **Export Selected**.

To upload a template deployment file:

1. Click **Choose File** and browse to and select the required JSON file.
2. Click **Upload Deployment File**.
3. If the template includes HAProxy SSL terminations, use the **Choose File** button to select the required SSL certificates to use.
4. For each Virtual Service to be created:
 - Ensure that the IP address is correct.
 - Decide if you'd like the IP address to be brought up on import, if not enable (check) the appropriate *Disable new IP address* checkbox.
 - Specify the associated Backend (Real) Servers. For layer 7 Real Servers, set the *Re-encrypt* checkbox as needed and if required configure the *Cert* and *CACert* for mTLS. Use the appropriate **+ Add Row** button to specify additional Real Servers.
5. Click **Apply Configuration**.

Disaster Recovery

To recover a single appliance, you'll need to restore from backup as described in the section above. For virtual and cloud based appliances, other hypervisor / cloud recovery methods can also be used.

To recover from a hardware failure that requires the firmware to be re-installed, please refer to [Firmware Recovery using a USB Memory Stick](#).

To recover a failed member of a HA pair the *Peer Recovery* method can be used, please refer to [Disaster Recovery After Node \(Primary or Secondary\) Failure](#).

Being Prepared - Creating Backups

To be able to quickly recover your appliance, it's important that you create regular configuration backups and keep them stored in a secure location. Ideally, you should keep a backup of both the Primary and Secondary appliances.

Firmware Recovery using a USB Memory Stick

If a hardware appliance has suffered a hardware failure that once resolved requires the firmware to be re-installed, follow steps 1 to 4 below.



Step 1 - Download the latest Appliance Disk Image

The latest disk image can be downloaded from our website - please contact support for more information.

Step 2 - Extract the Image from the Compressed Archive

Extract the image using tar under Linux or something like WinRar or 7-Zip under Windows (not the built-in Windows extractor).

Step 3 - Prepare the USB Stick

Using Linux

After formatting the USB stick run the following command:

```
dd if=/imagefilename.img of=/dev/name-of-usb-disk
```

e.g.

```
dd if=/tmp/v7.5.0_r3368.img of=/dev/sda
```

Note

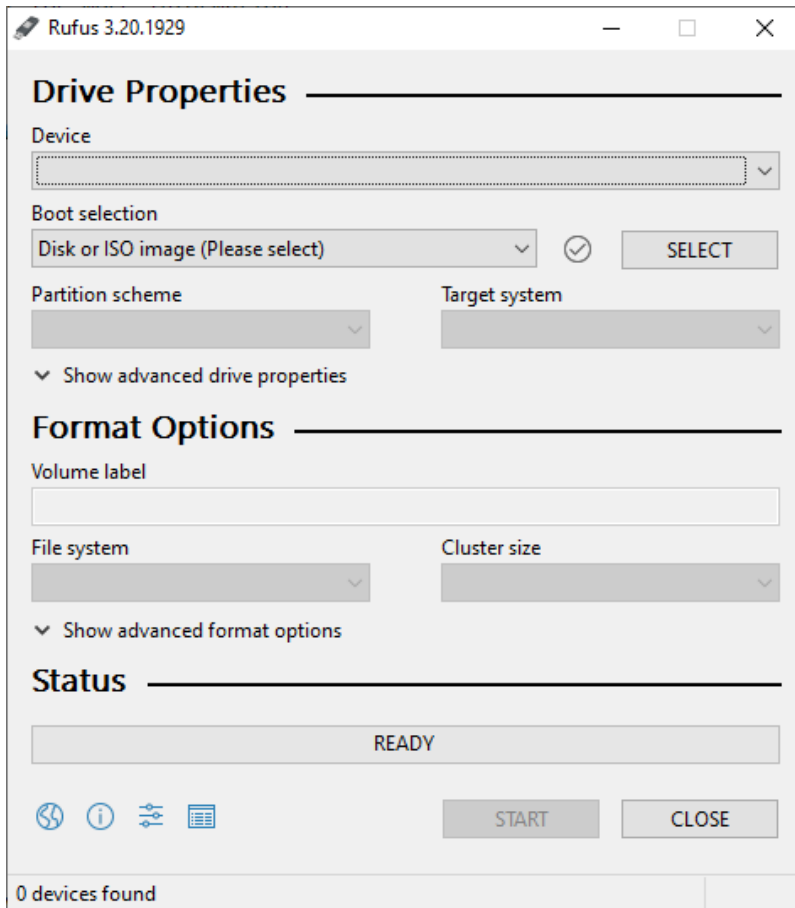
Do not use /dev/sdx where "x" is a number, for example - /dev/sda1 as this will install to a partition on your usb stick. Use the whole disk **/dev/sda** Instead.

Warning

Be careful - make sure you specify the correct disk!

Using Windows

For Windows, a third party image writer must be used. Several free options are available, the example below uses **Rufus** which can be downloaded [here](#). The download is an executable file.



Start Rufus, select the relevant USB device and the source image and click **START**.

 **Warning** Be careful - make sure you specify the correct disk!

Step 4 - Restore the Appliance Using the USB Stick

1. Boot the appliance into System Setup.
2. Ensure that UEFI boot is enabled and that the USB device is first in the boot order.

 **Note** From v8.7.2 all ISO images require UEFI to boot.

3. Boot the appliance, after the initial boot messages the following prompt will appear:

```
DO YOU WISH TO CONTINUE?  
Please enter yes or no  
Type "yes" and press <ENTER>
```

Once complete, the following message will be displayed:

```
Installation Finished
```

4. As directed, press any key to shutdown the appliance.
5. Once shutdown, remove the USB stick.
6. Power up the appliance.
7. Run through the [Network Setup Wizard](#) - this will enable a password for the "root" user to be set.
8. Login at the console:

Username: root

Password: <configured-during-network-setup-wizard>

9. Run the model specific command to ensure that the hardware is configured correctly.

 **Note** Please contact support for the command for your specific model.


10. Now run the following additional commands:


```
# lbrestore <ENTER>
# lbcleanboot <ENTER>
```

11. Reboot the appliance.
12. Run the [Network Setup Wizard](#) once again to configure the management IP address, other network settings and passwords for the "root" Linux account and the "loadbalancer" WebUI account.
13. Now re-apply your license key file to ensure the newly restored appliance is correctly licensed. Please contact support if you have any issues.

Disaster Recovery After Node (Primary or Secondary) Failure

For a clustered pair of load balancers, recovery of a failed node is quick and simple. The procedure is the same whether the Primary has failed or the Secondary has failed.

 **Note** Using the Peer Recovery feature means that the HA pair is re-established **without** disrupting any of the services that are currently running on the working appliance.

 **Note** Make sure that SSH Password access is enabled on the remaining operational node. This can be configured using the WebUI menu option: **Local Configuration > Security**, clearing the **Disable SSH Password Access** checkbox and clicking **Update**.

1. If the failed node is still on, power it down.
2. For a hardware appliance:
 - Disconnect all cables.
 - If the SSD/HD has failed and has been replaced and needs to be re-imaged, follow [these steps](#) to restore the appliance firmware.



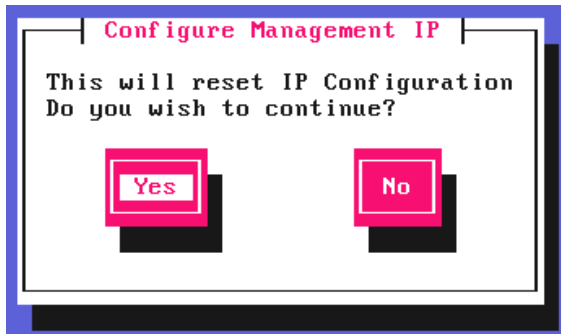
3. Power up the new/repaired/re-imaged appliance.

4. Login to the console as:

username: setup

password: setup

5. The Configure Management IP screen will be displayed:



6. Select **Yes** and hit <ENTER> to continue.

7. On the Peer Recovery screen you'll be asked if you're recovering from node failure:



8. Select **Yes** and hit <ENTER> to continue.

9. Now continue through the Network Setup Wizard to configure the initial network settings - ensure these are the same as the failed appliance. Continue until you reach the Peer Recovery screen.

 **Note**

For more information about configuring network settings, [Configuring Initial Network Settings](#).

Peer Recovery

Please enter the active Loadbalancers IP:

Peer Node IP 192.168.111.151

Done Back

10. Enter the IP address of the active appliance, select **Done** and hit <ENTER> to continue.

Peer Recovery

Please enter the active Loadbalancers WUI Port:

Peer Node Port (Default:9443) 9443

Done Back

11. Enter the WebUI port of the active appliance, select **Done** and hit <ENTER> to continue.

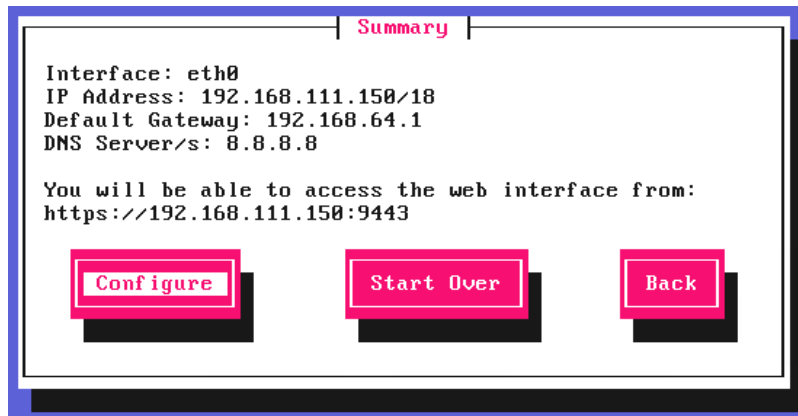
Peer Recovery

Please enter the password for the WUI loadbalancer user on the active peer

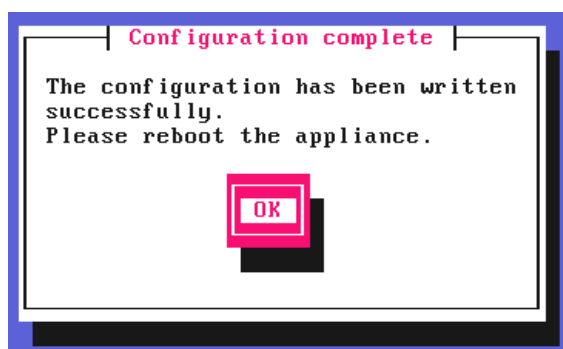
Peer Node WUI Password loadbalancer

Done Back

12. Enter the WebUI password for the "loadbalancer" user on the active appliance, select **Done** and hit <ENTER> to continue.



13. A summary of the local settings is displayed, hit <ENTER> to continue. The settings from the peer appliance will then be applied. Once the restore completes, the Configuration Complete message will be displayed:



14. To complete the restore and re-pairing process, reboot the newly restored appliance as mentioned in the on-screen message. Once rebooted, the HA pair will be re-synchronized and fully recovered.

The Primary's WebUI should now display:

Primary | Secondary

Active | Passive

[Link](#)

The Secondary's WebUI should now display:

Primary | **Secondary**

Active | **Passive**

[Link](#)

Chapter 15 - Technical Support

Introduction

Loadbalancer.org have a team of very experienced support engineers who are available to assist with your load balancer deployment.

Unlimited support is available as follows:

- During the cover period of any active support agreement

(to purchase a support package, please contact: sales@loadbalancer.org)

- During the 30 day Virtual Appliance trial period

(to download the trial, please go to: <https://www.loadbalancer.org/get-started/adc-free-trial/>)

WebUI Support Options

The WebUI's Support menu option includes 3 sub-options that are provided to assist when help is required.

Contact Us

This option provides details on how to contact the support team, the information we need to help us resolve the issue as quickly as possible and guidance on how to generate a technical support download which should also be included with your initial email.

Sending an email to support@loadbalancer.org creates a ticket in our help desk system and enables all technical support staff to view the case. This is the most efficient way to contact support and guarantees that any reported issues will be acted upon and addressed as quickly and efficiently as possible.

Technical Support Download

This option enables the Support Download to be created. The download is a compressed archive containing all log files and configuration files from the appliance and should be attached to your email.



Technical Support Download

When contacting **Loadbalancer.org Support**, you may be asked to supply the load balancer's configuration and log files. This page generates an archive of all the required files, which can then be downloaded to your PC.

Please click the button below to generate the archive.

The load balancer will collect the configuration files and logs into a compressed archive.

Download will begin shortly after clicking the button.

Send the archive by email to **Loadbalancer.org support**. If this is your first contact with support on this issue, please include your company name and details of the problem you are experiencing.

☒ **Do not include GZ files.**(This can decrease archive size in some circumstances)

Note: Generating the archive may take several minutes on a load balancer with extensive log files.

Generate Archive

Please click the button above to start the process.

- To minimize the size of the support download, **GZ** files (archived logs) are excluded by default. If these must be included in the archive, clear the checkbox shown above
- To generate the archive, click the **Generate Archive** button

Once downloaded, attach the file to your email when contacting support.

If the file is large, it can be uploaded using our upload server once a support ticket has been opened.

Useful Links

This option presents a number of self explanatory web links.

Remote Support

Where necessary, our support team may arrange a remote session to assist with trouble shooting. **Zoom** is the preferred tool and is used where possible.

Live Chat

Online chat can be invoked directly from the WebUI.

To enable online chat:

1. Using the WebUI, navigate to: **Local Configuration > Live Chat**.
2. Ensure that **Enable Live Chat** is enabled (checked).
3. Click Update.

To invoke online chat:

1. Using the WebUI, navigate to: **Live Chat**.



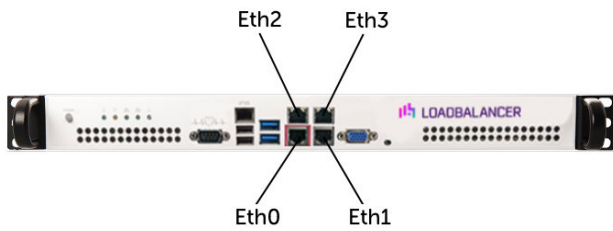
2. Enter the question you have.



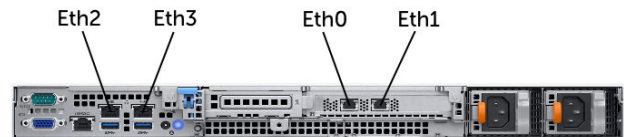
Appendix

Front & Rear Panel Layouts

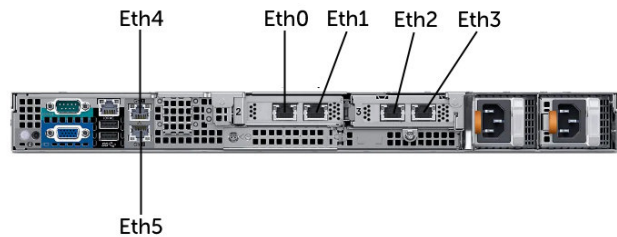
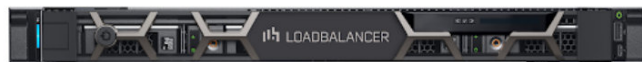
Enterprise Prime



Enterprise Flex



Enterprise Max



* The number of interfaces depends on your choice of interface cards

IPMI Configuration

The Enterprise Prime includes Intelligent Platform Management Interface (IPMI) which provides remote management and control.

IPMI Network Settings

IPMI can be accessed via the dedicated IPMI Ethernet interface or via one of the standard Ethernet interfaces in shared/bridged mode.

To use the dedicated IPMI interface, ensure that a network cable is plugged into the interface before powering up the appliance.

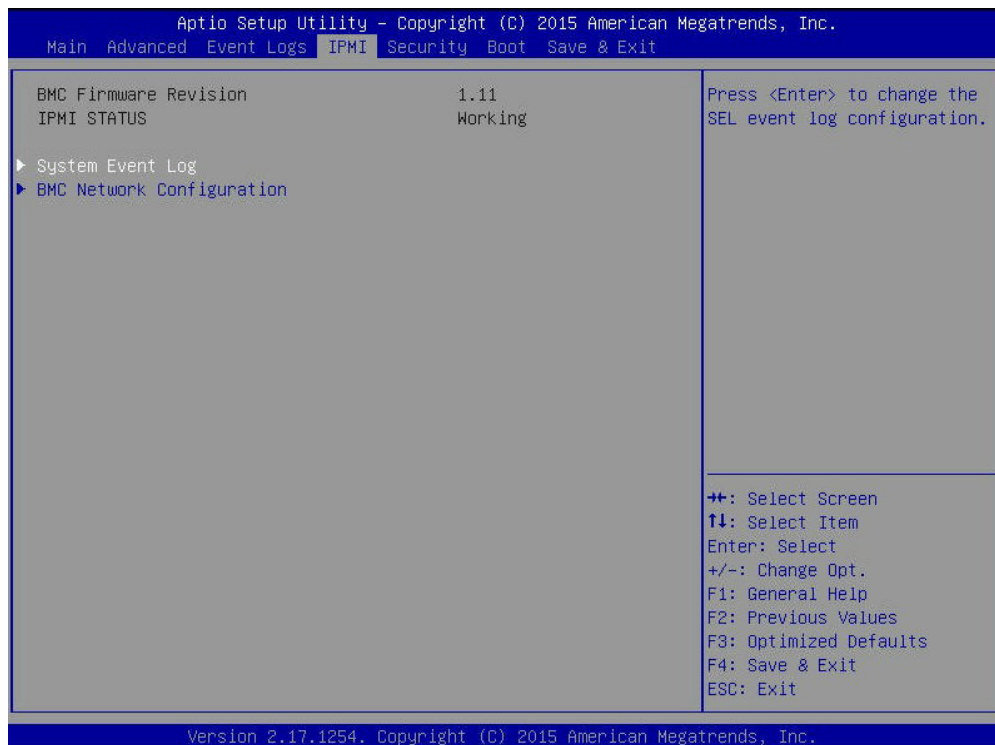
By default, the appliance attempts to use DHCP to obtain an IP address for IPMI. If a DHCP server is not available, the IP address must be set manually.



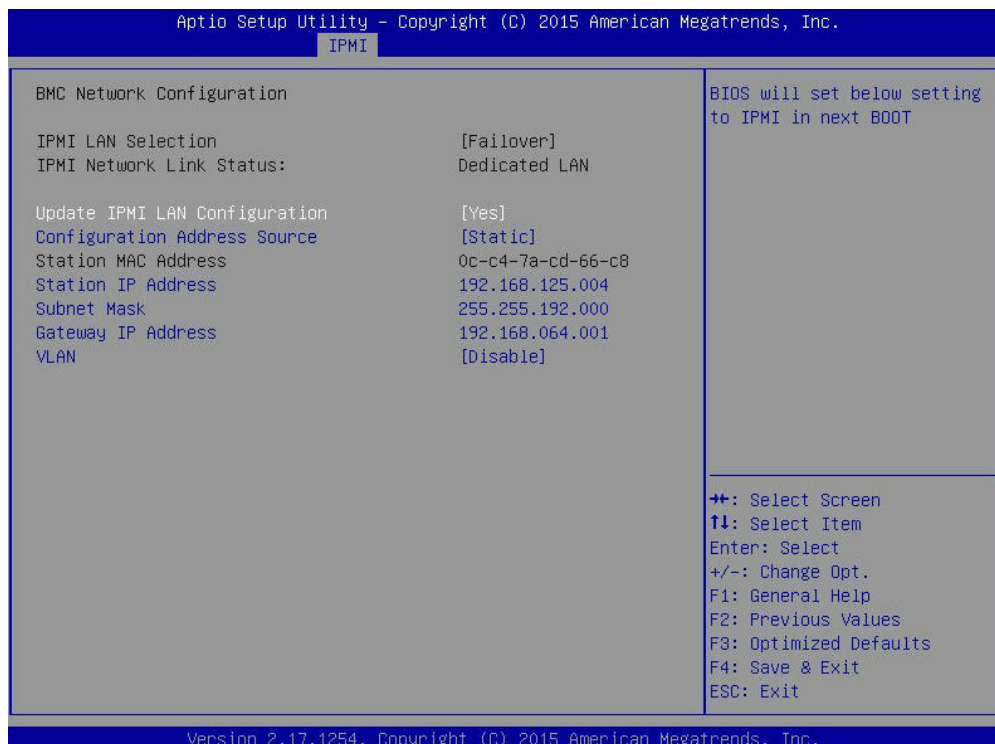
A static IP address can be assigned using the boot mode Setup Utility or by using the IPMI Web Interface.

Using the Setup Utility

To access the IPMI configuration menu via the Setup Utility, hit **** at boot time, then select the **IPMI** menu:



Select the **BMC Network Configuration** option and change **Update IPMI LAN Configuration** to **Yes**, all network settings can then be configured:



Note

Depending on the particular IPMI version, the option to change the IPMI LAN interface may be



greyed out as shown above. In this case, use the IPMI Web Interface (see below) to configure this setting.

To configure the IP address and other network settings:

1. Set **Configuration Address Source** to **Static**.
2. Configure the **IP address**, **Subnet Mask** and other network settings as required.

To exit the IPMI configuration menu, hit **<ESC>**.

To save any changes, select the **Save & Exit** menu, then select **Save Changes and Reset** to save and apply the new settings.

Using the IPMI Web Interface

Once the network is configured, the IPMI Web Interface can be accessed using a web browser:

```
https://<IPMI IP address>
```

The default user credentials to access the IPMI Web Interface are:

Username : ADMIN
Password : refer to the label on the appliance / packaging

Note

For older appliances, the password was set to "ADMIN".

Once logged in, the IPMI Web Interface can be used to monitor and configure the system, configure all IPMI settings and access the Remote Console.

Technical Support / More Information

If you have any questions, please contact support@loadbalancer.org.

For more information, refer to the [IPMI User Guide](#).

iDRAC Configuration

All Dell based hardware appliances shipped since January 2024 include an integrated Dell Remote Access Controller (iDRAC) Enterprise license which enables remote monitoring and remote control of the appliance. Details of all supported iDRAC features are available [here](#).

iDRAC Network Settings

iDRAC can be accessed via the dedicated iDRAC interface on the rear panel or by sharing one of the built in LAN On Motherboard (LOM) interfaces. Using the dedicated port ensures that iDRAC management traffic is separated from production workload.

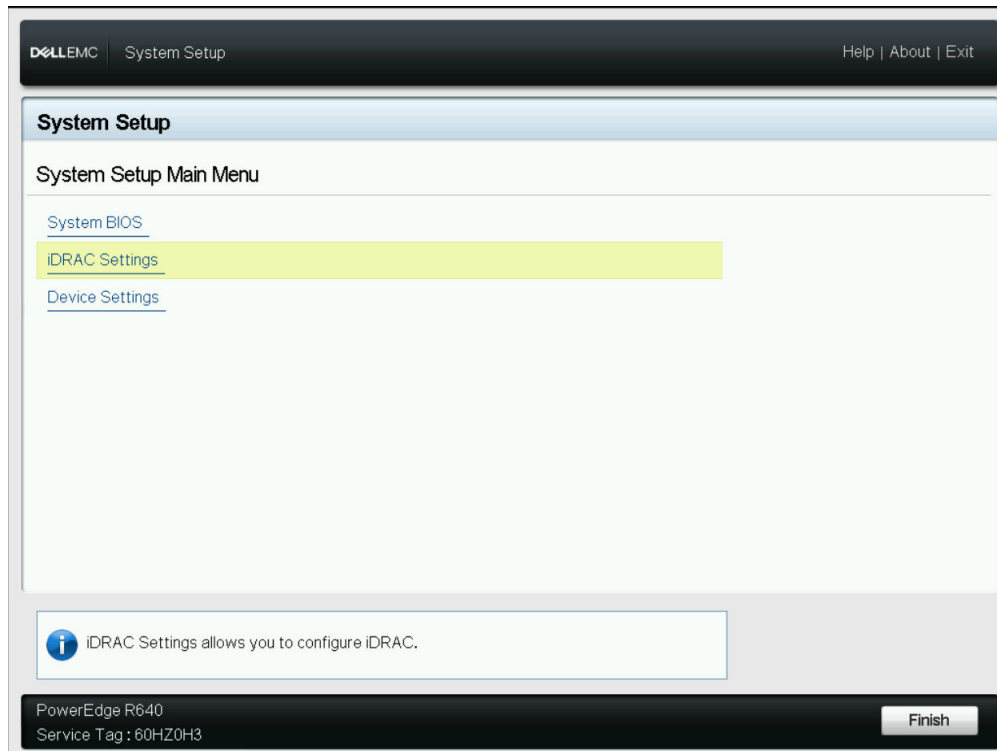


By default, the appliance attempts to use DHCP to obtain an IP address. If a DHCP server is not available, the IP address is set to **192.168.0.120/255.255.255.0**.

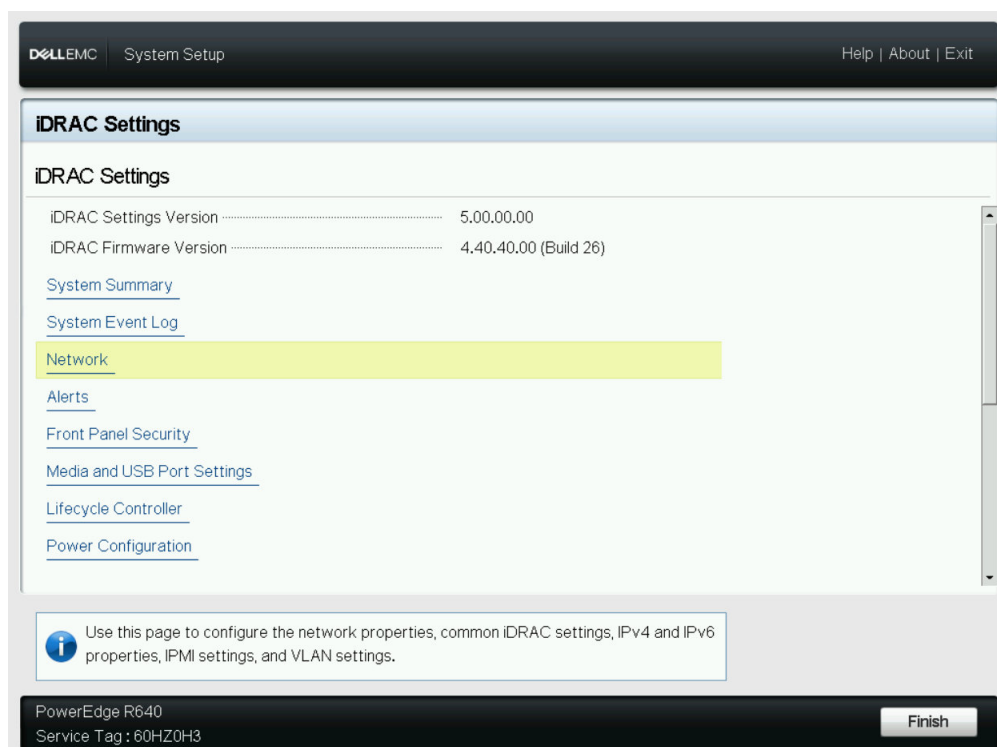
A static IP address can be assigned using System Setup or by using the iDRAC Web Interface.

Using System Setup

To access the iDRAC configuration menu via System Setup, hit **<F2>** at boot time, then select **iDRAC Settings**:



Select the **Network** option:



The iDRAC network can be enabled/disabled and the interface to use can be configured in the **NETWORK SETTINGS** section:

The screenshot shows the 'iDRAC Settings' window with the 'Network' tab selected. Under 'NETWORK SETTINGS', the 'Enable NIC' option is set to 'Enabled'. Other settings include 'NIC Selection' set to 'Dedicated', 'Failover Network' set to 'None', 'MAC Address' as 'D0:8E:79:BD:A1:16', 'Auto Negotiation' set to 'On', 'Auto Dedicated NIC' set to 'Disabled', 'Network Speed' set to '1000 Mbps', 'Active NIC Interface' set to 'Dedicated', and 'Duplex Mode' set to 'Full Duplex'. Under 'COMMON SETTINGS', 'Register DRAC on DNS' is set to 'Disabled'. A help box at the bottom states: 'Select Enabled to enable NIC. When NIC is enabled, it activates the remaining controls in this group. When a NIC is disabled, all communication to and ... (Press <F1> for more help)'. The footer shows 'PowerEdge R640' and 'Service Tag : 60HZ0H3' with a 'Back' button.

Scroll down to the **IPV4 SETTINGS** & **IPV6 SETTINGS** section to configure the IP address, subnet mask and other network settings:

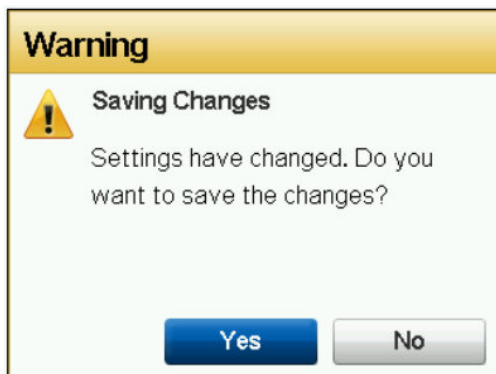
This screenshot shows the 'iDRAC Settings' window with the 'Network' tab selected, displaying the 'IPV4 SETTINGS' and 'IPV6 SETTINGS' sections. In the 'IPV4 SETTINGS' section, 'Enable IPv4' is set to 'Enabled', 'Enable DHCP' is set to 'Disabled', 'Static IP Address' is '192.168.125.12', 'Static Gateway' is '192.168.64.1', 'Static Subnet Mask' is '255.255.192.0', 'Use DHCP to obtain DNS server addresses' is set to 'Disabled', 'Static Preferred DNS Server' is '192.168.64.1', and 'Static Alternate DNS Server' is '0.0.0.0'. In the 'IPV6 SETTINGS' section, 'Enable IPv6' is set to 'Disabled' and 'Enable Auto-configuration' is set to 'Enabled'. A help box at the bottom states: 'Displays the current NIC mode.'. The footer shows 'PowerEdge R640' and 'Service Tag : 60HZ0H3' with a 'Back' button.

Note

For additional information, see [this Dell article](#).

To exit the iDRAC configuration menu, click **Back** and click **Finish**. Click **Finish** again to exit System Setup.

If changes were made, you'll be prompted to save them:



Click **Yes** to save the changes.

Using the iDRAC Web Interface

Once an iDRAC IP address is assigned, the iDRAC Web Interface can be accessed using a web browser:

```
https://<iDRAC IP address>
```

The default user credentials to access the iDRAC Web Interface are:

```
Username : root
Password : refer to the pull-out tab in the front of the appliance
```

Once logged in, the iDRAC Web Interface can be used to monitor and configure the system, configure all iDRAC settings and access the Virtual Console.

Technical Support / More Information

If you have any questions, please contact support@loadbalancer.org.

For more information, refer to the [iDRAC User Guide](#).

Appliance IPv4 Address Format (CIDR notation)

When specifying IP addresses on the appliance, CIDR format is used. The following table shows the various masks and the corresponding IPv4 IP/CIDR equivalents:

Mask	IP/CIDR Prefix
255.255.255.255	a.b.c.d/32
255.255.255.254	a.b.c.d/31
255.255.255.252	a.b.c.d/30
255.255.255.248	a.b.c.d/29



Mask	IP/CIDR Prefix
255.255.255.240	a.b.c.d/28
255.255.255.224	a.b.c.d/27
255.255.255.192	a.b.c.d/26
255.255.255.128	a.b.c.d/25
255.255.255.000	a.b.c.d/24
255.255.254.000	a.b.c.d/23
255.255.252.000	a.b.c.d/22
255.255.248.000	a.b.c.d/21
255.255.240.000	a.b.c.d/20
255.255.224.000	a.b.c.d/19
255.255.192.000	a.b.c.d/18
255.255.128.000	a.b.c.d/17
255.255.000.000	a.b.c.d/16
255.254.000.000	a.b.c.d/15
255.252.000.000	a.b.c.d/14
255.248.000.000	a.b.c.d/13
255.240.000.000	a.b.c.d/12
255.224.000.000	a.b.c.d/11
255.192.000.000	a.b.c.d/10
255.128.000.000	a.b.c.d/9
255.000.000.000	a.b.c.d/8
254.000.000.000	a.b.c.d/7
252.000.000.000	a.b.c.d/6
248.000.000.000	a.b.c.d/5
240.000.000.000	a.b.c.d/4
224.000.000.000	a.b.c.d/3
192.000.000.000	a.b.c.d/2
128.000.000.000	a.b.c.d/1

Appliance Configuration Files & Locations

The various configuration files used by the appliance are listed in the table below:

Configuration	File & Location
Network	/etc/sysconfig/network-scripts/ifcfg-eth*
Firewall	/etc/rc.d/rc.firewall
Firewall Lockdown Wizard	/etc/rc.d/rc.lockdownwizard.conf
System XML File	/etc/loadbalancer.org/lb_config.xml
Layer 4	/etc/ha.d/conf/loadbalancer.cf
Layer 7	/etc/haproxy/haproxy.cfg
Layer 7 (manual)	/etc/haproxy/haproxy_manual.cfg
Pound SSL	/etc/pound/pound.cfg
STunnel SSL	/etc/stunnel/stunnel.conf
SSL Certificates	/etc/loadbalancer.org/certs
Heartbeat	/etc/ha.d/ha.cf
Heartbeat Resources	/etc/ha.d/haresources
GSLB	/opt/polaris/etc/polaris-lb.yaml
GSLB Topology	/opt/polaris/etc/polaris-topology.yaml
WAF	/etc/httpd/waf.conf.d/90-wafs.conf
SNMP	/etc/snmp/snmpd.conf



Visit us: www.loadbalancer.org

Phone us: +44 (0)330 380 1064

Phone us: +1 833 274 2566

Email us: info@loadbalancer.org

Follow us: [@loadbalancer.org](https://twitter.com/loadbalancer.org)

About Loadbalancer.org

Loadbalancer.org's mission is to ensure that its clients' businesses are never interrupted. The load balancer experts ask the right questions to get to the heart of what matters, bringing a depth of understanding to each deployment. Experience enables Loadbalancer.org engineers to design less complex, unbreakable solutions - and to provide exceptional personalized support.

