MANUALS

Appliance Administration Manual

Version 8.6.3 Revision 1.0.0

loadbalancer.org

Table of Contents

Chapter 1 - Introduction

About this Manual

This document covers all required administration information for v8.6.x Loadbalancer.org appliances.

About the Appliance

The Loadbalancer.org appliance runs the GNU/Linux operating system with a custom kernel configured for load balancing.

The core software is based on customized versions of Centos 6.x/RHEL 6.x, Linux 4.9.x (cloud appliance kernel versions may be different), LVS, HA-Linux, HAProxy, Pound, STunnel & Ldirectord. 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).

Appliances can be deployed as single units 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

The latest version of the appliance (v8.6.3) includes the following new features, improvements, bug fixes and security updates:

New Features

None in this release.

Improvements

None in this release.

Bug Fixes

- Fixed issue where fallback server might be inaccessible when using TProxy.
- Fixed warning generated in web interface when layer 7 ACL rules are added via LBCLI.
- Fixed issue where editing health check scripts could break a cluster.

Security Updates

Upgraded OpenSSL to version 1.1.1n.

Appliance Configuration Overview

Initial network configuration is carried out at the console using the Network Setup Wizard. Once an IP address has been allocated, load balanced services can be configured using the WebUI; either using the Setup Wizard (for

Layer 7 services) or by manually defining the Virtual Services (VIPs) and associated Real Servers (RIPs).

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 should be fully configured first, then the Secondary appliance should be added to create an HA pair. Once the HA pair is configured, load balanced services will be automatically synchronized from the Primary to Secondary appliance. Load balanced services should then be configured & modified on the Primary appliance and the Secondary 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, 2 private IPs must be specified – one for the VIP when it's active on the Primary and one 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 all 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:

- **Secure - (default)** - in this mode:
  - the WebUI is accessible on HTTPS port **9443.** If you attempt to access the WebUI on HTTP port **9080** you will be redirected to HTTPS port **9443**
  - access to the *Local Configuration > Execute shell command* menu option is disabled
  - the ability to edit the firewall script & the lockdown wizard is disabled
  - 'root' user console & SSH password access are disabled
- **Custom** - in this mode, the security options can be configured to suit your requirements
- **Secure - Permanent** - this mode is the same as **Secure** but the change is ***irreversible***

Important	Only set the security mode to **Secure - Permanent** if you are 100% sure this is what you want!

To configure the Security Mode:

1. Using the WebUI, navigate to: *Local Configuration > Security*.
2. Select the required *Appliance Security Mode* - if **Custom** is selected, configure the additional options according to your requirements.
3. Configure the *HTTPS Port for Web User Interface*, *Web Interface SSL Certificate* and *Ciphers to use* according to your requirements.

4. Click **Update**.

Passwords

The password for the '**loadbalancer**' WebUI user account and the '**root**' Linux user account are set during the Network Setup Wizard. These can be changed at any time.

1 - the 'root' Linux account

As explained above, 'root' user console & SSH password access are disabled by default. If enabled, the 'root' password can be changed at the console, or via an SSH session using the following command:

```
# passwd
```

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 defined during instance deployment, run the following command:

```
$ sudo su
```

2 - 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
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	9080	WebUI - HTTP (disabled by default)
TCP	9081	Nginx fallback page
TCP	9443	WebUI - HTTPS

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
- Configuration Guide - Amazon AWS
- Configuration Guide - Microsoft Azure
- Configuration Guide - Google Cloud Platform

Contacting Support

This manual should provide you with enough information to be very productive with your Loadbalancer.org appliance. However, if there are aspects of the appliance that have not been covered, or if you have any questions, 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 support 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.

Load Balancing Algorithms

The Loadbalancer.org appliance supports several different load balancing algorithms. Each one has its 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 recieve 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, 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*** (more information is available here) has to be solved, for layer 4 NAT mode, the default gateway on the Real Servers 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, additional configuration steps can be taken to force the client IP to be logged. This includes using TProxy or 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?

Note	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 the current user sessions will be transferred to other servers in the cluster without the users needing to re login to the application again. **If your application doesn't have a persistent data store then you can't have seamless fail over for your backend servers**.

Do your web servers store persistent information on local drives?

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

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

Replication Solutions for Shared Data

On UNIX you can use the RSYNC command to replicate files, on Windows Server you can use RSYNC as well but you may prefer ROBOCOPY that's included by default in newer versions of Windows Server or in the resource kit for older versions. Usually you will upload your content to one Primary server and then replicate it 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 user will need to re-log in and possibly lose shopping baskets etc.

This problem is easily resolvable by implementing a shared persistent data store for the cluster. This is usually either done with a shared backend database or a shared memory solution.

Persistence (aka Affinity)

Persistence is a feature that is required by many web applications. Once a user has interacted with a particular server all subsequent requests are sent to the same server thus persisting to that particular server. It is normally required when the session state is stored locally to the web server as opposed to a database.

What do You do if Your Application is not Stateless?

Some applications require state to be maintained such as:

- Terminal Services/Remote Desktop Services
- SSH
- FTP (upload)
- SMTP (incoming)

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 based on source IP address. You lose the ability to have transparent fail-over, but you do still get increased capacity and manageability. This persistence problem occurs with all load balancers and all vendors use standard methods and technologies to mitigate the issue.

Loadbalancer.org Persistence Options

The following default persistence options are available:

- Source IP (subnet)
- Cookie (Active or Passive)
- SSL session ID
- X-Forwarded-For header

- Microsoft Connection Broker/Session Broker Integration

Note

It's also possible to define other custom persistence types if required using the manual configuration option available for layer 7 Virtual Services. For more information, please refer to Layer 7 - Custom Configurations.

The standard layer 4 persistence method is source IP persistence, you can handle millions of persistent connections at layer 4. Just modify your Virtual Service to be persistent if you require source IP persistence.

Cookies are a layer 7 based persistence method that can offer more even traffic distribution and also handle any clients where the source IP address may change during the session (e.g. mega proxies).

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 effectively breaks the persistence.

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. If the objective is clear and measurable, you know when you have achieved your goal.

Load balancers have a number of flexible features and benefits for your technical infrastructure and applications.

The primary question to consider is: **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.
Reliability	Running an application on one server gives you a single point of failure. Utilizing a load balancer moves the point of failure to the load balancer. At Loadbalancer.org we always advise that you deploy load balancers as clustered pairs to remove this single point of failure. For more information on configuring an HA pair 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 your 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.
Secondary	The normally passive appliance in a HA Clustered Pair.
VIP	Virtual IP address - the address of the load balanced cluster of RIPs, the address presented to connecting clients.

RIP	The Real IP address of a backend server in the cluster.
Floating IP	The Floating IP Address is automatically created whenever a VIP is configured, the FIP 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, descriptive term for a network device that can route packets based on TCP/IP header information.
Layer 7	Part of the seven layer OSI model, descriptive term for a network device that can read and write the entire TCP/IP header and 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 using two physical network cards or by assigning two addresses to one physical network card.
GW	The Default Gateway for a backend server in the cluster.
Eth0	Usually the internal Ethernet interface, although this is optional. Also known as Gb0 on the Enterprise 10G, 50G and 100G.
Eth1	Usually the external Ethernet interface, although this is optional. Also known as Gb1 on the Enterprise 10G, 50G and 100G.
Eth2	Third Ethernet interface.
Eth3	Fourth Ethernet interface.
Eth4	Fifth Ethernet interface (Enterprise 100G only, also depends on choice of interface cards).
Eth5	Sixth Ethernet interface (Enterprise 100G only, also depends on choice of interface cards).

What is a Virtual IP Address?

Most load balancer vendors use the term Virtual IP address (VIP) to describe the address that the cluster is accessed from. It's important to understand that the Virtual IP address (VIP) refers to both the physical IP address and also to the logical load balancer configuration. Likewise the real IP (RIP) address 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 services can 'float' (move) between Primary and Secondary appliances.

What is a Floating IP Address?

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

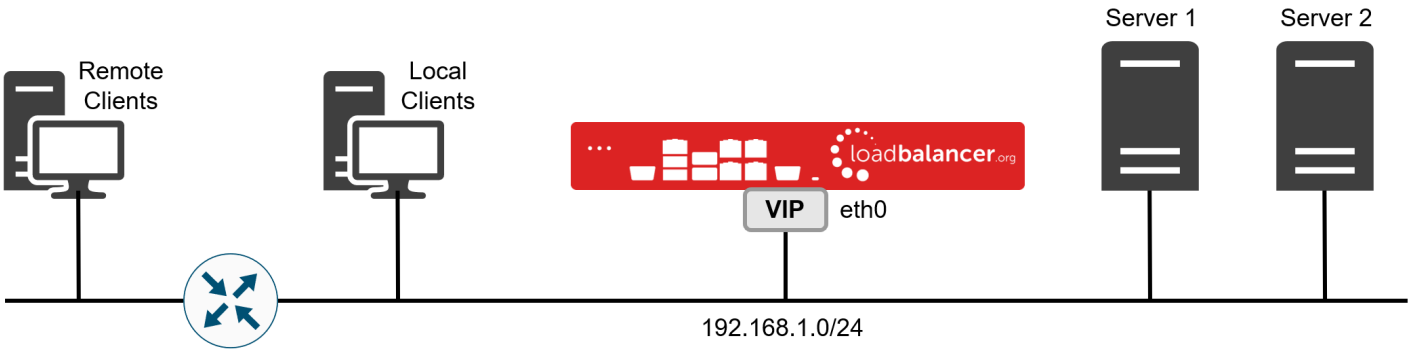
Chapter 3 - Topologies & Load Balancing Methods

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

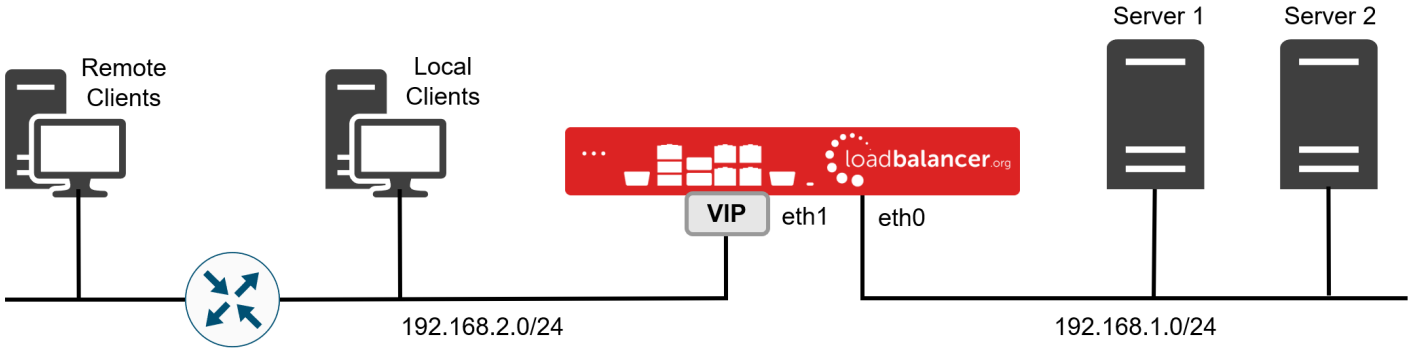
The VIP and the load balanced servers are located in a single subnet. The load balancer requires a single network interface adapter - eth0 in the diagram below.



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 is one of the most flexible load balancers available. The design allows different load balancing modules to utilize the core high availability framework of the appliance. Multiple load balancing methods can be used at the same time or in combination with each other. The table below describes the methods supported by the appliance.

Layer	Method	Comments	Topology	Note
Layer 4	DR (Direct Routing)	Ultra-fast local server based load balancing • Requires the ***ARP Problem*** to be solved on each Real Server - for more details please refer to DR Mode Considerations	One-Arm (*)	1
Layer 4	NAT (Network Address Translation)	Fast Layer 4 load balancing • The appliance must be the default gateway for the Real Servers	One or Two-Arm	1
Layer 4	TUN	Similar to DR but works across IP encapsulated tunnels	One-Arm	2
Layer 4	SNAT (Source Network Address Translation)	Fast layer 4 SNAT supporting both TCP & UDP • Very simple to implement • Requires no Real Server configuration changes	One or Two-Arm	3
Layer 7	SSL Termination (STunnel & Pound)	Usually required in order to process cookie persistence in HTTPS streams on the load balancer • SSL Termination is processor intensive	One or Two-Arm	4
Layer 7	SNAT (Source Network Address Translation using HAProxy)	Layer 7 allows greater flexibility including full SNAT and remote server load balancing, cookie insertion and URL switching • Very simple to implement • Requires no Real Server configuration changes • Not as fast as Layer 4 methods	One or Two-Arm	4

(*) 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

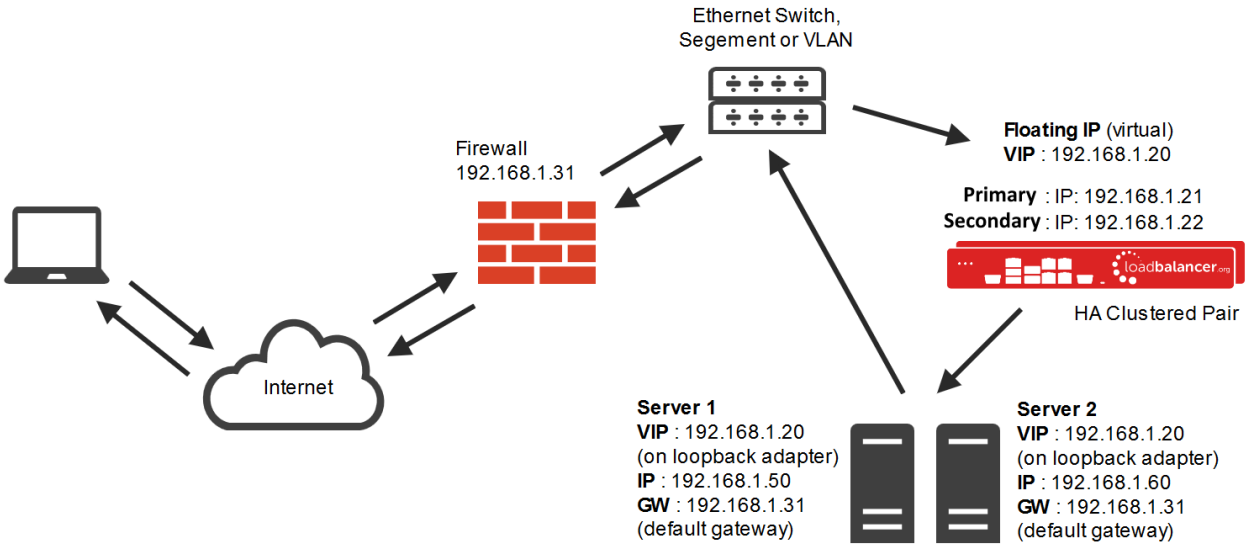
1. Recommended for high performance fully transparent and scalable solutions.
2. Only required for Direct Routing implementation across routed networks (rarely used).
3. Recommended when you want to load balance both TCP and UDP but you're unable to use DR mode or NAT mode due to network topology or Real Server related reasons.
4. Recommended if HTTP cookie persistence is required, also used for several Microsoft applications such as

Exchange, Sharepoint & Remote Desktop Services and for overall deployment simplicity since Real Servers can be on any accessible subnet and no Real Server changes are required.

Layer 4 DR Mode

One-arm direct routing (DR) mode is a very high performance solution that requires little change to your existing infrastructure.

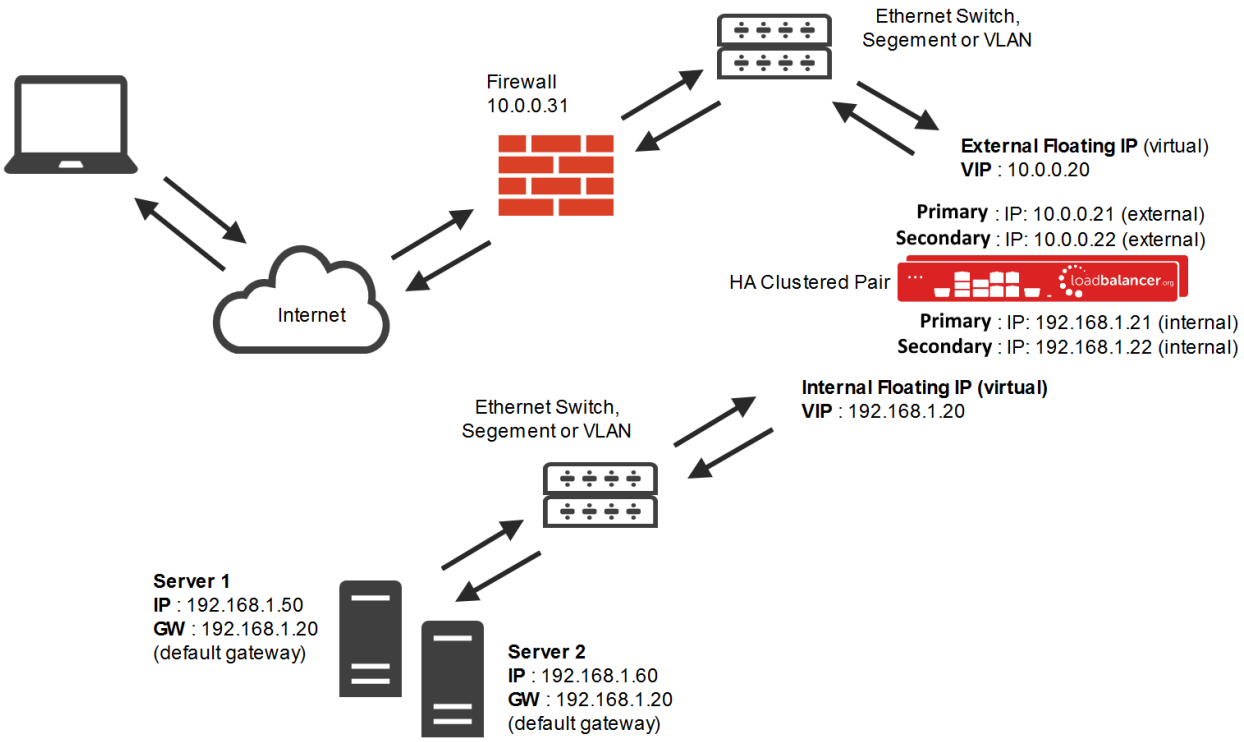
Note | Kemp, Brocade, Barracuda & A10 Networks call this *Direct Server Return* and F5 call it *N-Path*.



- 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 you need to ensure that the Real Server (and the load balanced application) respond to both the Real Server's own IP address and the VIP.
- The Real Servers should not respond to ARP requests for the VIP. Only the load balancer should do this. Configuring the Real Servers 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 for HTTP, 50 times quicker for Terminal Services and much, much faster for streaming media or FTP.
- The load balancer must have an Interface in the same subnet as the Real Servers to ensure layer 2 connectivity required for DR mode to work.
- 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

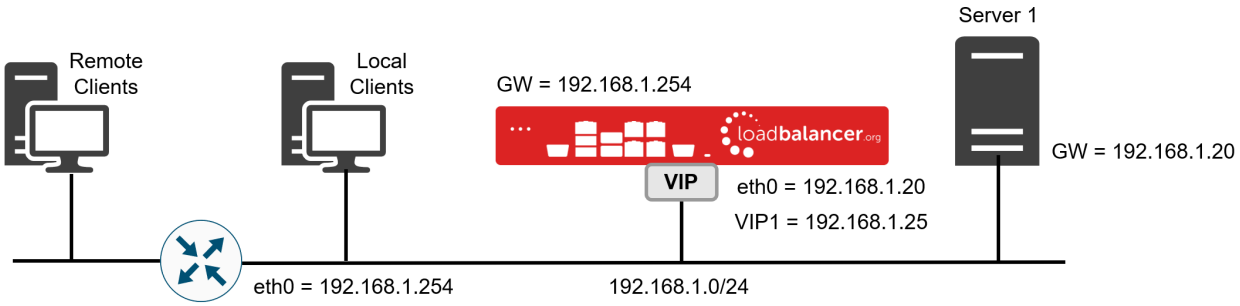
Layer 4 NAT mode is a high performance solution, although not as fast as layer 4 DR mode. This is because real server responses must flow back to the client via the load balancer rather than directly as with DR mode.



- The 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. If the Real Servers require Internet access, *Autonat* 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.
    - 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. SMTP or 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:

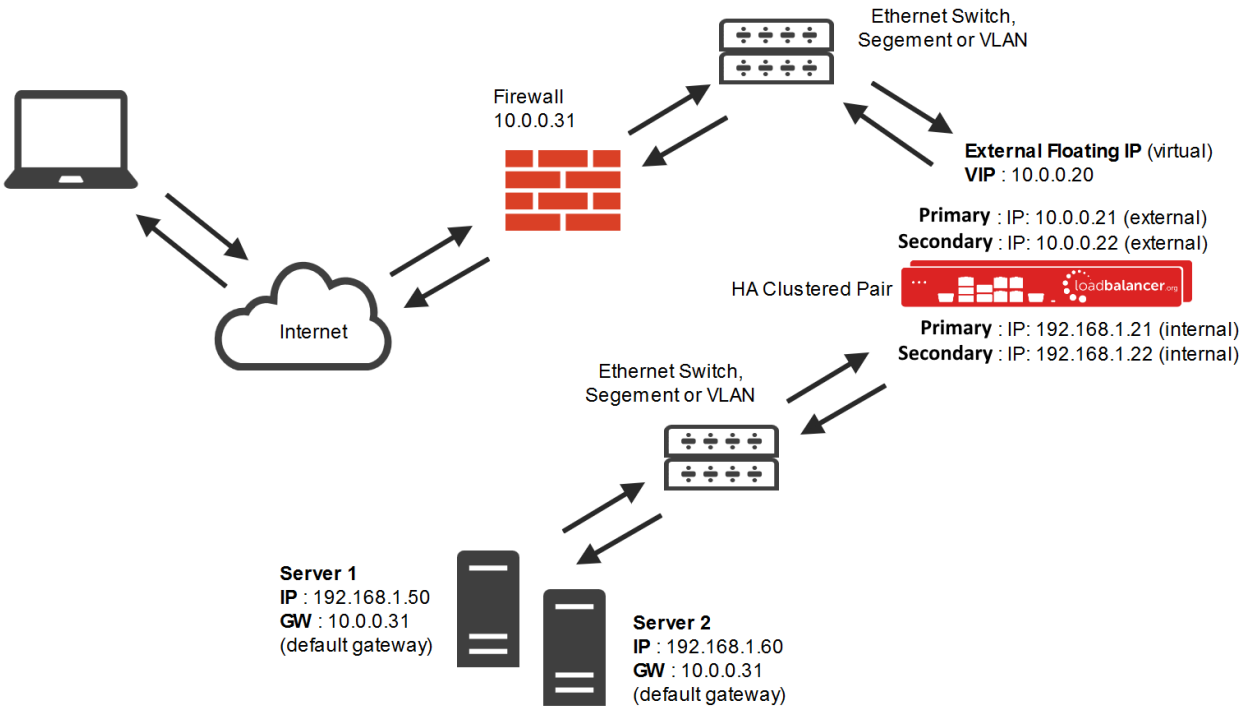
Source	192.168.1.50:80	Destination	x.x.x.x:34567

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

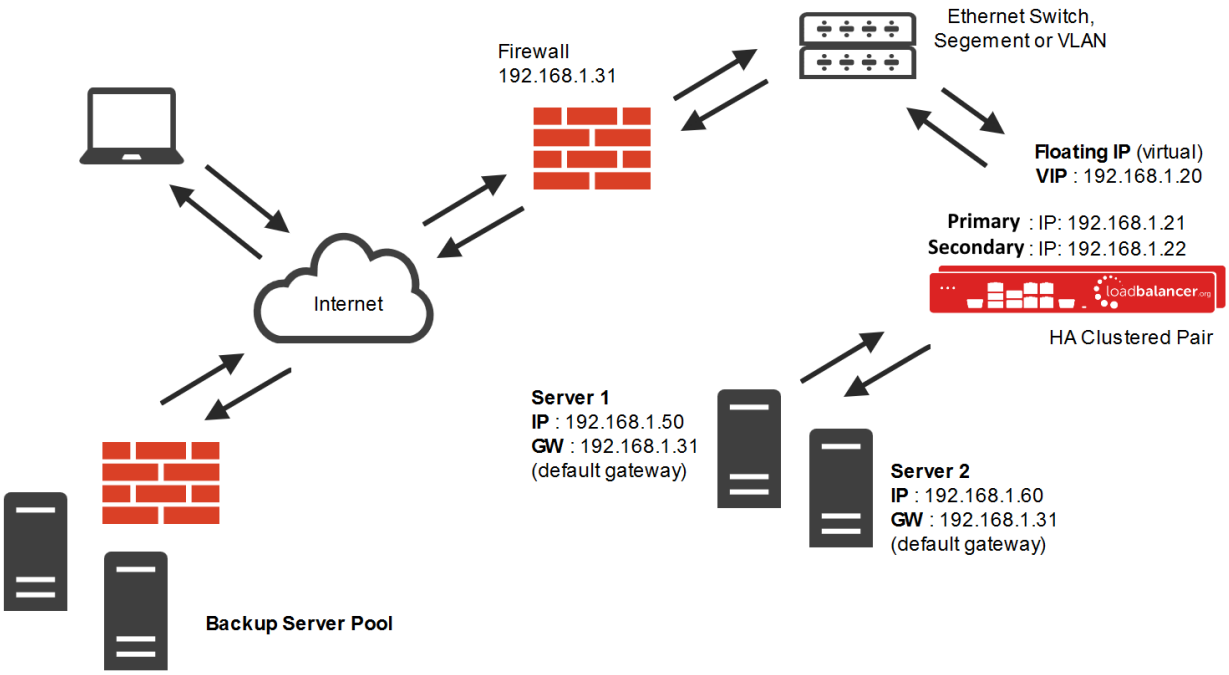
Layer 4 SNAT mode is a high performance solution, although not as fast as Layer 4 NAT mode or Layer 4 DR mode.



- The load balancer translates all requests from the external Virtual Service to the internal Real Servers in the same way as NAT mode - please refer to Layer 4 NAT Mode for more information.
- 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, eth0 is normally used for the internal network and eth1 is used for the external network although this is not mandatory.
- If the Real Servers require Internet access, Autonat should be enabled using the WebUI option: *Cluster Configuration > Layer 4 - Advanced Configuration*, the external interface should be selected.
- Requires no additional configuration changes to the load balanced Real Servers.
- Port translation is not possible with Layer 4 SNAT mode, e.g. VIP:80 → RIP:8080 is not 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.



- Because layer 7 SNAT mode is a full proxy, any server in the cluster can be on any accessible subnet 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, eth0 is normally used for the internal network and eth1 is used for the external network although this is not mandatory.
- Requires no additional 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 11 - 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 as described here. 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 works very well with either Pound or STunnel when SSL termination is required. It also enables content switching and header manipulation rules to be implemented. It does not require any changes to the Real Servers and can be deployed in one-arm or two-arm mode. HAProxy is a high performance solution, but since it operates as a full proxy it cannot perform as fast as the layer 4 solutions. Layer 7 SNAT mode is non-transparent by default, i.e. the Real Servers will see the source IP address of the load balancer. 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 (i.e. the backend servers) make sure that Windows **NLB** (Network Load Balancing) is **completely disabled** to ensure that this does not interfere with the operation of the load balancer.

Chapter 4 - Appliance Fundamentals

The Hardware Appliance - Unpacking and Connecting

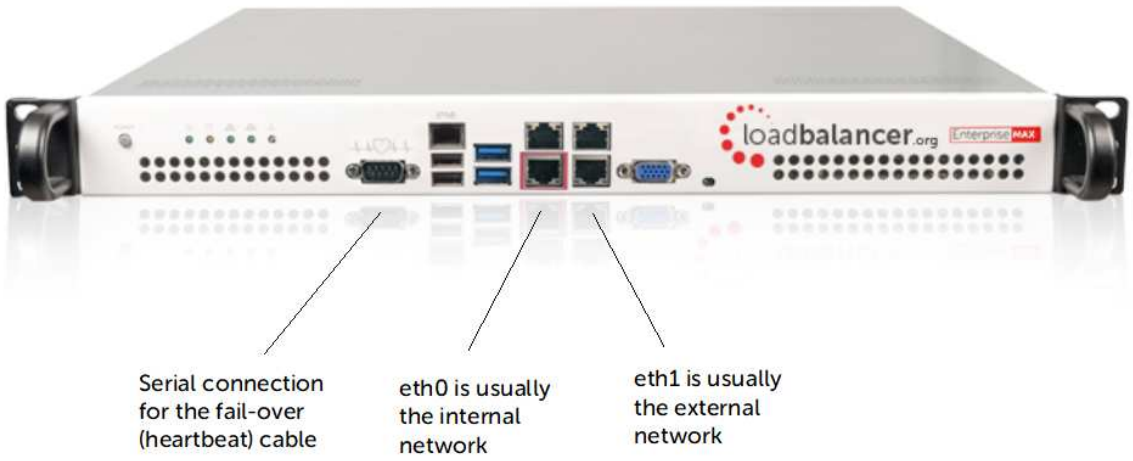
1. Remove all packaging and rack mount the appliance if required.
2. Connect the power lead from the power socket to the mains or UPS.

> 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. 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 units must be able to communicate either via 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 to the VGA port and keyboard to one of the USB ports.
6. Check mains power is on and press the power switch to start the appliance. The fans should start & front panel LEDs should light.



> Note | The above image shows the Enterprise 1G. For information on other models please refer to Front & Rear Panel Layouts.

The Virtual Appliance - Hypervisor Deployment

Supported Hypervisors

Currently, the Virtual Appliance is available for the following hypervisors:

- VMware ESXi : v4.0 & later
- Virtual Box : v4.0 & later

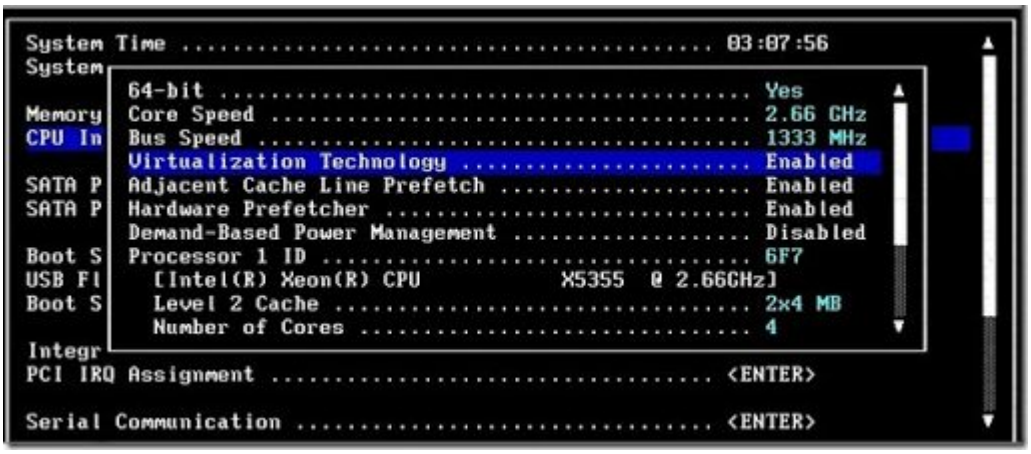
- Microsoft Hyper-V : v2012 & later
- KVM : Kernel version v2.6.20 & later
- XEN : v6.0 & later
- Nutanix AHV

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 or AMD-V compliant CPUs

For an Intel based server, VT must be enabled in the BIOS as shown in the example below:



If your server is unable to support 64bit guests, an error message will be displayed when attempting to start the VA.

Once deployed, the VA is allocated the following resources by default:

- 2 vCPUs
- 4GB RAM
- 20GB disk

The CPU and memory allocations are suitable for a PoC or for low 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.

Downloading the Appliance

All downloads are accessible from the following location: https://www.loadbalancer.org/get-started/. To access the downloads, enter your name (optional), email address, phone number (optional), and specify the application that you'll be load balancing (optional), then select the Hypervisor type and click **Download Now.** The various download links will then be presented on screen and we'll also send you an email containing the same links. Once the

required version is downloaded, extract the archive using your preferred utility. Each download also includes a *ReadMe.txt* file which explains the VA deployment process.

Note	All information provided is 100% confidential. We may follow up with an email or phone call to see how you're getting on with the trial 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	For the VA, 4 NICs are included but only eth0 is connected by default at power on. If the other NICs are required, these should be connected using the network configuration screen within the Hypervisor.

VMware Deployment

The steps required depend on which VMware environment is in use.

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.

vSphere Client

1. Right-click any inventory object that is a valid parent object of a virtual machine, such as a data center, folder, cluster, resource pool, or host, and select **Deploy OVF Template**.
2. Select the **Local File** option, click **Choose Files**, browse to the VA download location and select both the **OVF** and **VMDK** files.
3. Complete the remaining options according to your requirements and deploy the VA.

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

VMware tools are pre-installed on the appliance which enables basic console control functions such as power on/off etc. The installed version of the various kernel modules and drivers is controlled by Loadbalancer.org at build time to ensure that only stable, fully tested versions are deployed. If the tools are later upgraded, these drivers and modules may be over-written. Therefore we do not recommend a full tool re-installation. If you do want

to update the basic tool functionality (i.e. without affecting the installed drivers and modules) please follow the steps listed in our blog.

Hyper-V Deployment

Windows 2008 R2

1. Start Hyper-V Manager, then using the right-click menu or the Actions pane select *Import Virtual Machine* and then click **Next**.
2. Browse to the location of the extracted download and select the folder LBVMHYPER-Vv8.
3. Select the option "*Copy the virtual machine (create a new unique ID)*" and also select the "*Duplicate all files so the same virtual machine can be imported again*" checkbox, click **Import**.
4. The import will start, once complete the new appliance will appear in the Virtual Machine list.
5. The appliance has 4 NIC cards, to connect these right-click the appliance and select *Settings* then for each Network Adapter select the required network.
6. Right-click and select **Start** to power up the appliance, allow a minute to boot.
7. If you're deploying a clustered pair, you'll first need to do one of the following steps before importing the second virtual machine. If this is not done, the second virtual machine cannot be deployed because the disk from the first import already exists, and there will therefore be a conflict:
    1. Shutdown the first VM and modify the name of the disk, ***Or***
    2. Change the default file location using the Hyper-V *Settings* option in the *Actions* pane.

Once one of the above has been done, repeat steps 1-6 to create the second virtual machine.

Windows 2012 and Later

1. Start Hyper-V Manager, then using the right-click menu or the Actions pane select *Import Virtual Machine* then click **Next**.
2. Browse to the location of the extracted download and select the folder LBVMHYPER-V3v8.
3. Click **Next** until prompted for the Import Type, make sure that '*Copy the virtual machine (create a new unique ID)*' is selected and click **Next**.
4. Tick the checkbox '*Store the Virtual Machine in different location*', then define a suitable location for the virtual machines files and click **Next**.
5. Define a location for the virtual hard disk files.
6. Click **Next**, then click **Finish** to complete the import process. Once complete, the load balancer will appear in the Virtual Machines list.
7. The appliance has 4 NIC cards, to connect these right-click the appliance and select Settings then for each Network Adapter select the required network.
8. Highlight the new load balancer and start it either by using the right-click menu or the Actions pane.

If you're deploying a clustered pair, repeat steps 2-8 for the Secondary unit, making sure that a different folder location is selected in steps 4 & 5.

Linux Integration Services

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

KVM Deployment

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 Deployment

For detailed installation and deployment guidance, please refer to our Nutanix blog.

XEN Deployment

The following steps should be followed on the XEN host:

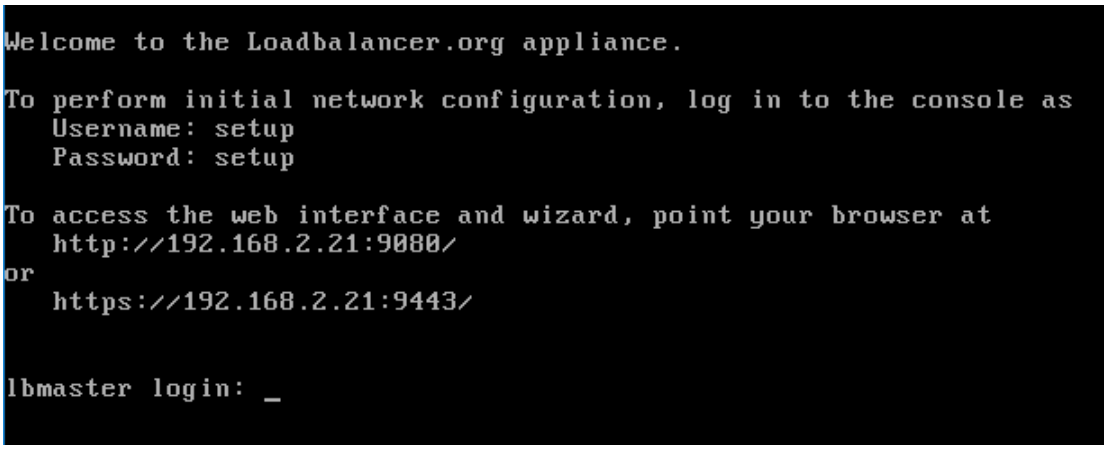
1. Extract the archive.
2. Import the **xva** file into XEN.

Cloud Appliance Deployment

For details of our cloud based products, please refer to the relevant quick start guide in the documentation library.

Configuring Initial Network Settings

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

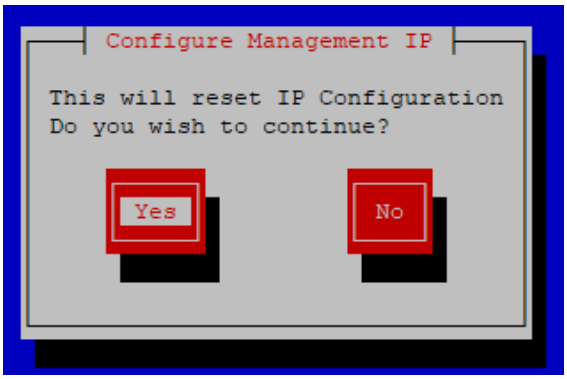
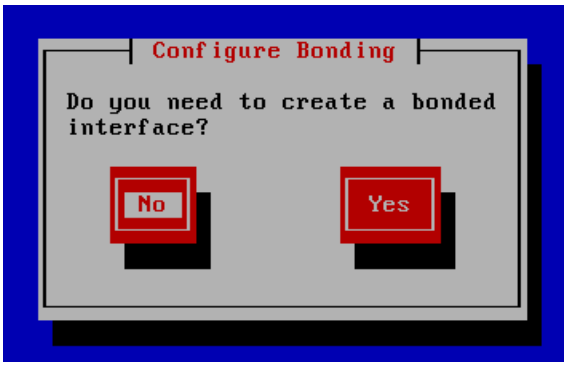
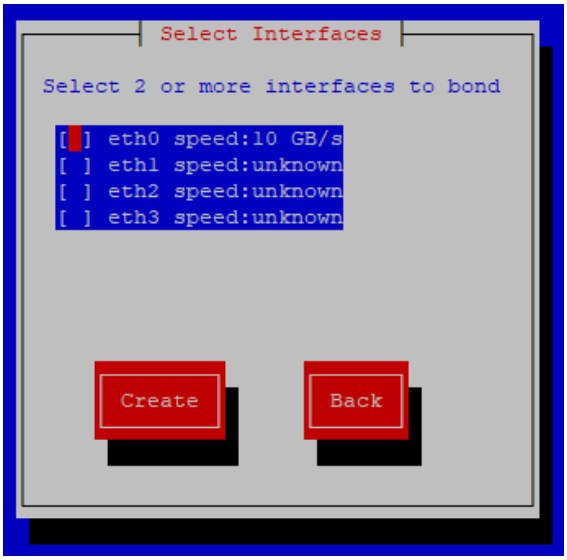


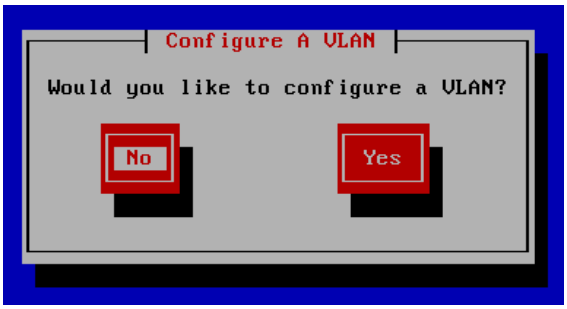
As mentioned in the text, to perform initial network configuration, login as the 'setup' user at the appliance console. Once logged in, the Network Setup Wizard will start automatically.

login to the console:

Username: setup
Password: setup

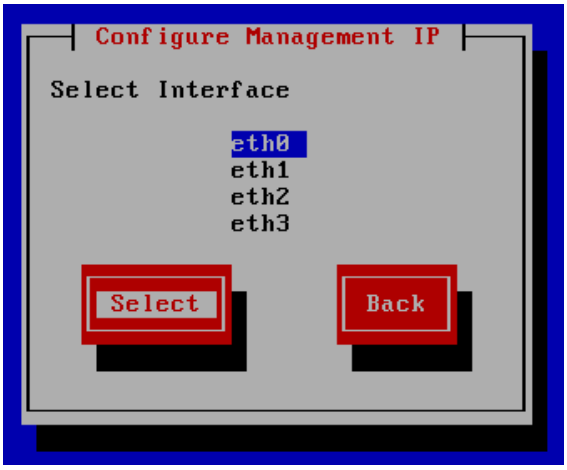
A series of screens will be displayed that allow network settings to be configured:

	To continue with the Network Setup Wizard select **Yes** and hit <ENTER> to continue.
	A list of available interfaces will be shown, hit <ENTER> to continue.
	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 screen shown below will be displayed:
	Using the space bar, select the interfaces you'd like to include in the bond, then click **Create**.

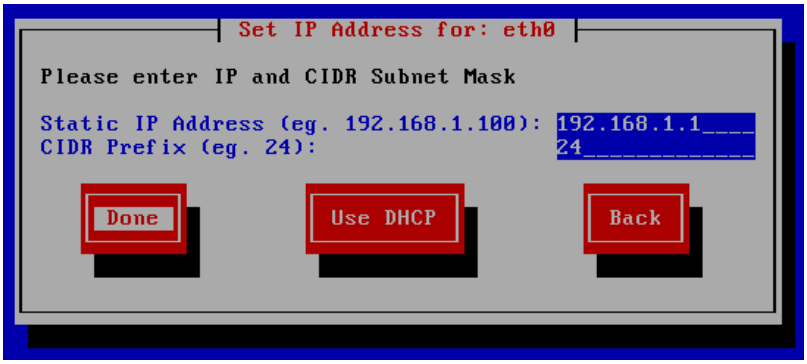


Select **Yes** If you want to configure a VLAN, if not leave **No** selected, then hit <ENTER> to continue.

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



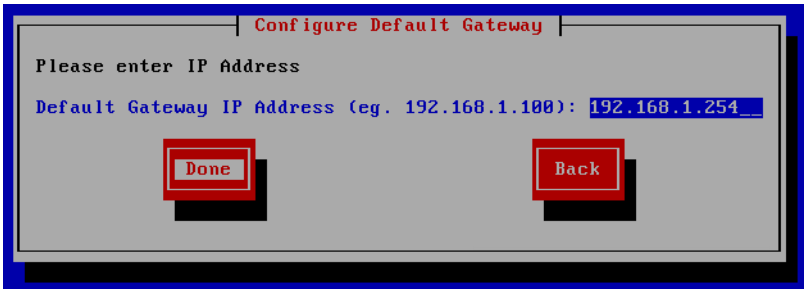
Select the interface that will be used to manage the appliance, select **Select** and hit <ENTER> to continue.



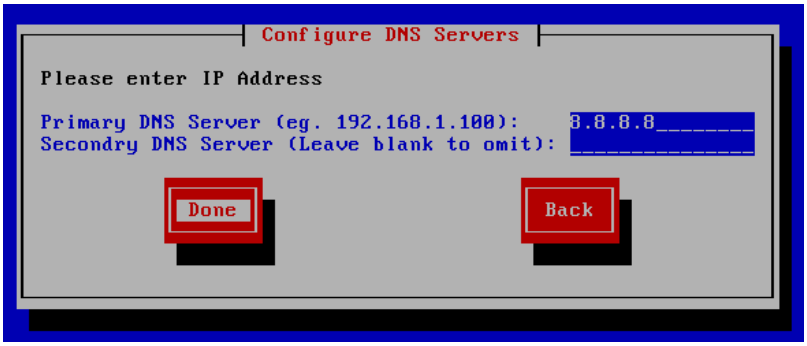
Either enter the required management IP address & CIDR prefix and select **Done** or select **Use DHCP** to request an address and then hit <ENTER> to continue.

Note

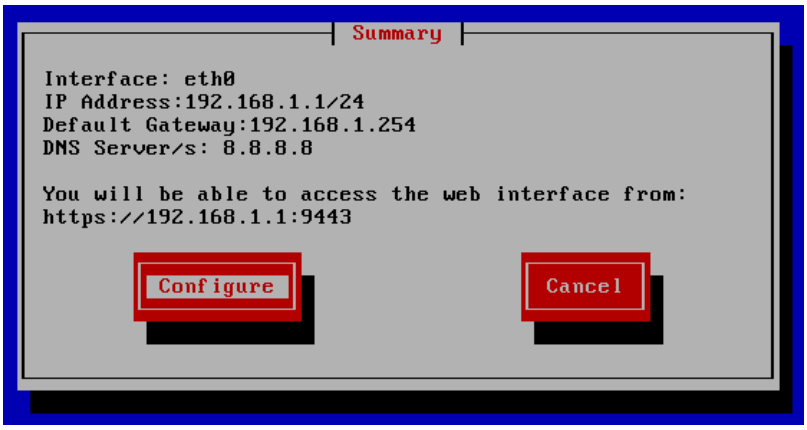
A subnet mask such as 255.255.255.0 is not valid, in this case enter 24 instead.



Enter the default gateway address or, select **Done** and hit <ENTER> to continue.



Define the required DNS server(s), select **Done** and hit <ENTER> to continue.

	A summary of all settings is displayed, if everything looks good hit <ENTER> to continue, all settings will then be applied.
	Hit <ENTER> to continue.
	Enter the password you'd like to use for the 'loadbalancer' WebUI user account and the 'root' Linux user account, select **Done** and hit <ENTER> to continue.

At this stage you'll be asked if you're recovering from node (i.e. Primary or Secondary) failure.

If you're simply deploying a new appliance, select **No** 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.

Appliance Access & Configuration Methods

The appliance can be accessed & configured both locally and remotely.

Local Methods

Console Access

For a VA, use the Hypervisor's UI to access the load balancer's console. For a hardware appliance, simply connect a monitor and keyboard to the load balancer, power up and you'll be presented with a login prompt. The console can also be accessed via the serial port if the default heartbeat configuration is used, i.e. heartbeat is configured to communicate over the network only.

Log in to the console:

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**, enable the required option(s) and click **Update**.

Appliance Configuration using Links

Once logged into the console, the text based Links browser can be used. To start Links and bring up the text based administration interface, use the following command:

```
links 127.0.0.1:9080
```

Log in to Links:

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

Use the *Up*, *Down* & *Enter* keys to move between and select the various menu options, then scroll to the bottom of the screen to view the associated settings. Follow the onscreen instructions to modify values.

Note

Configuring the appliance using Links is not practical and is not recommended. It can be useful for minor changes when a full web browser is not at hand, but not for a full configuration. Appliance configuration should be carried using the WebUI.

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.

Remote Methods

When configuring the appliance remotely, take care when changing network and firewall settings. If you do lock yourself out, you'll either need local console access or you can use remote management tools such as IPMI or iDRAC. The Enterprise 1G supports IPMI, the Enterprise 10G, 50G and 100G support iDRAC. For more information on configuring IPMI please refer to IPMI (Remote Management) Configuration, for more information on configuring iDRAC please refer to iDRAC (Remote Management) Configuration.

The appliance can be remotely accessed using the following tools:

- HTTP/HTTPS Web Browser - for Web User Interface (WebUI) access
- OpenSSH (Linux hosts) or PuTTy (Windows hosts) - for secure shell access
- OpenSCP (Linux hosts) or WinSCP (Windows hosts) - for secure file transfer

Accessing the WebUI

The WebUI is accessed using a web browser. By default, user authentication is based on local Apache .htaccess files. User administration tasks such as adding users and changing passwords can be performed using the WebUI menu option: *Maintenance > Passwords*.

Note

A number of compatibility issues have been found with various versions of Internet Explorer and Edge. The WebUI has been tested and verified using both Chrome & Firefox.

Note

If required, users can also be authenticated against LDAP, LDAPS, Active Directory or Radius. For more information please refer to External Authentication.

1. Using a browser, access the WebUI using the following URL:

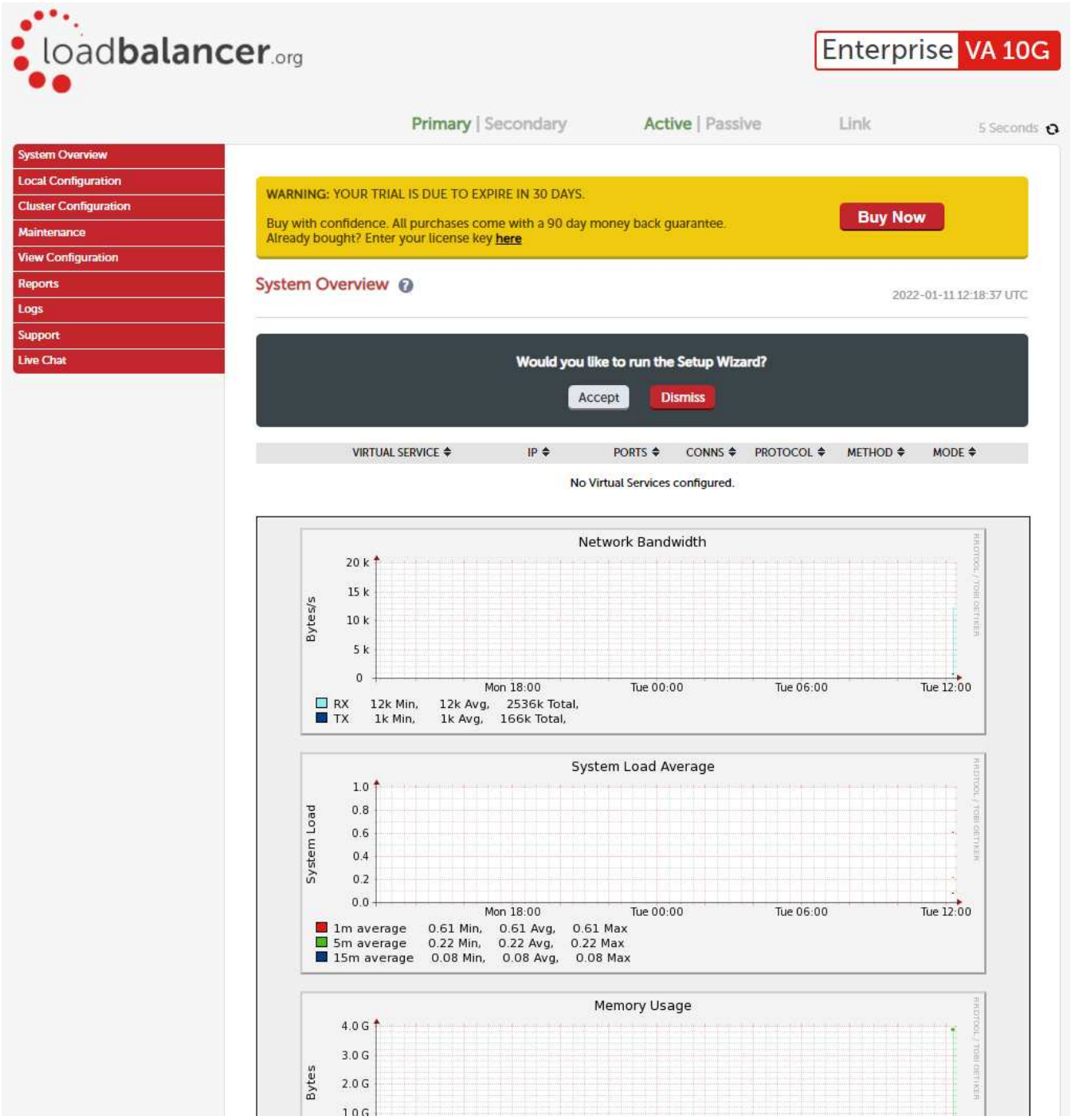
   https://<IP-address-configured-during-network-setup-wizard>:9443/lbadmin/

2. Log in to the WebUI:

   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:



Note | The WebUI for the VA is shown, the hardware and cloud appliances are very similar. The yellow licensing related message is platform & model dependent.

3. You'll be asked if you want to run the Setup Wizard. If you click **Accept** the Layer 7 Virtual Service configuration wizard will start. If you want to configure the appliance manually, simple click **Dismiss**.

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 taking 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

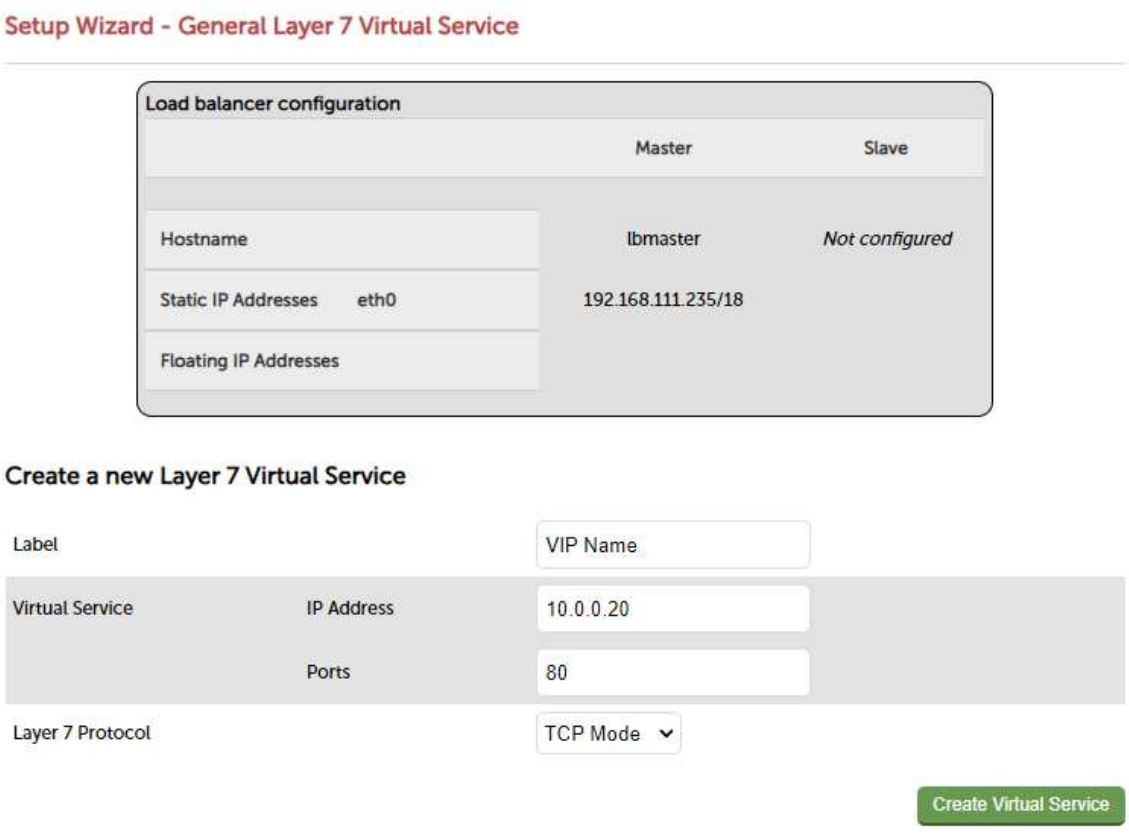
Configuring Load Balanced Services using the Wizard

The wizard can be used to setup one or more Layer 7 Virtual Services and associated Real Servers. Layer 4 services must be configured manually.

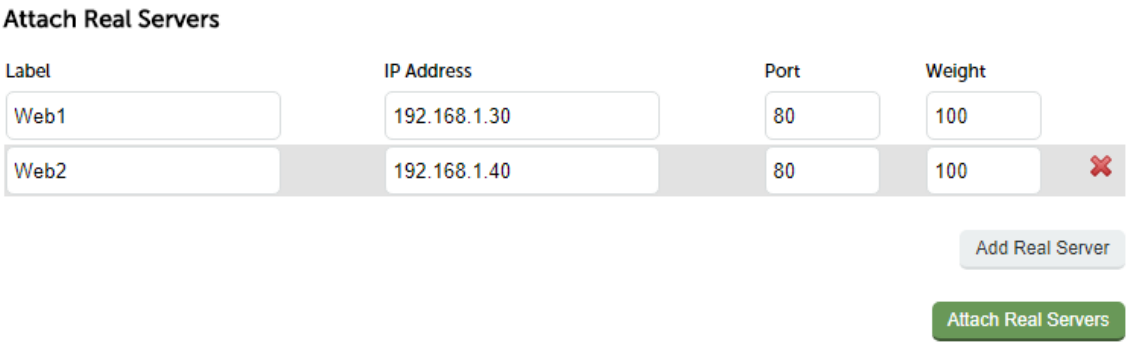
First, set the Appliance's IP address as described in the section: Configuring Initial Network Settings.

To run the wizard:

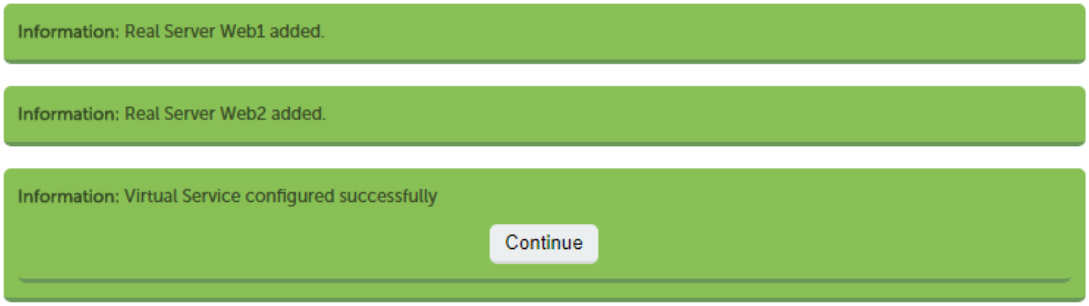
1. Open the WebUI and start the wizard by clicking the **Accept** button shown above, or by using the WebUI menu option: *Cluster Configuration > Setup Wizard* and clicking **General Layer 7 Virtual Service**.
2. Define the required Virtual Service settings as shown in the example below:



3. Click **Create Virtual Service**.
4. Now continue and add the associated load balanced servers (Real Servers) as shown below:



- Use the **Add Real Server** button to define additional Real Servers and use the red cross to delete Real Servers.
- Once you're happy, click **Attach Real Servers** to create the new Virtual Service & Real Servers.
- A confirmation message will be displayed as shown in the example below:



5. Click **Continue**.
6. Finally, reload HAProxy using the **Reload HAProxy** button in the blue 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 defined.
Note	To restore manufacturer's settings use the WebUI menu option: *Maintenance > Backup & Restore > Restore Manufacturer's Defaults*. This will reset the IP address to 192.168.2.21/24.
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.

Configuring Load Balanced Services Manually

To configure the appliance manually using the WebUI, please refer to Chapter 6 - Configuring Load Balanced Services.

Chapter 5 - Appliance Management

Network Configuration

Physical Interfaces

The Enterprise 1G, Enterprise 10G and all virtual models have 4 network interfaces. The Enterprise 50G also has 4 interfaces; 2 of these can be customized to suit your connectivity requirements when purchased. The Enterprise 100G supports up to 6 interfaces depending on your choice of interface cards when purchased. Comprehensive information on all models is available here.

For the VA, only the first interface is connected by default, the other interfaces can be connected when required using the Hypervisor's management interface. If multiple logical interfaces are required, these can be added simply by specifying multiple IP addresses as shown below. If multiple cables must be connected, 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 (layer 4) or two-arm SNAT mode (layer 7) highly available configuration.

Configuring IP Addresses

As mentioned in the previous chapter, initial network settings can easily be configured using the Network Setup Wizard. For more information on using the wizard please refer to Configuring Initial Network Settings.

IP addresses can also be configured using the WebUI menu option: *Local Configuration > Network Interface Configuration*. If a single interface is required, *eth0* is typically used. If 2 interfaces are required, *eth0* is typically used as the internal interface and *eth1* is used as the external interface. However, unlike other appliances on the market you can use any interface for any purpose.

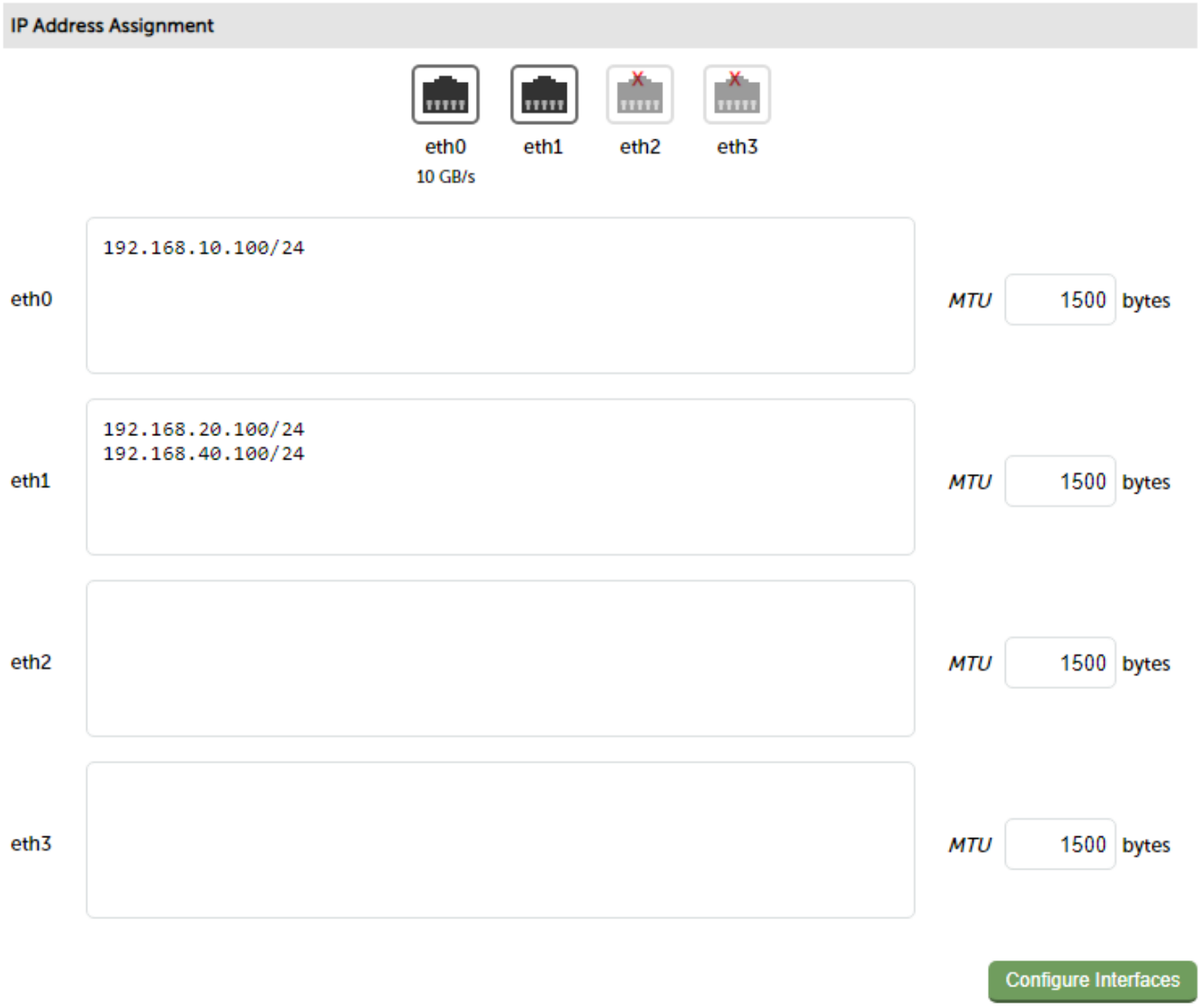
In a simple one-arm configuration, you would just need to configure the IP address and subnet mask for one interface, e.g. *eth0* and if there are remote clients, the relevant default gateway. Both IPv4 and IPv6 addresses can be configured.

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, then 192.168.2.100/24 would be entered in the relevant interface field as shown in the example below:

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. Assign the required IP address/mask, multiple addresses can be assigned as shown below:



3. Click **Configure Interfaces**

Note	If you already have Virtual Services defined when making changes to the network configuration, you should verify that your Virtual Services are still up and working correctly after making the changes.

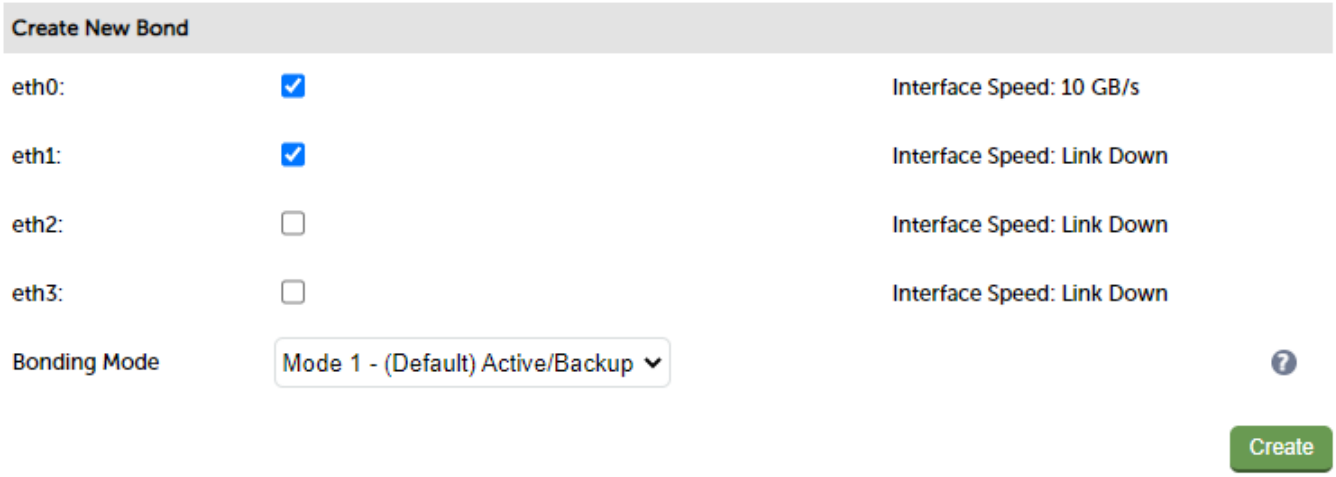
Note	For the VA, four NICs are included but only eth0 is connected by default at power on. If the other NICs are required, these should be connected using the network configuration screen within the Hypervisor.

Configuring Bonding

The appliance supports bonding of multiple network interfaces.

To Configure Bonding:

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



3. Select the required *Bonding Mode:*

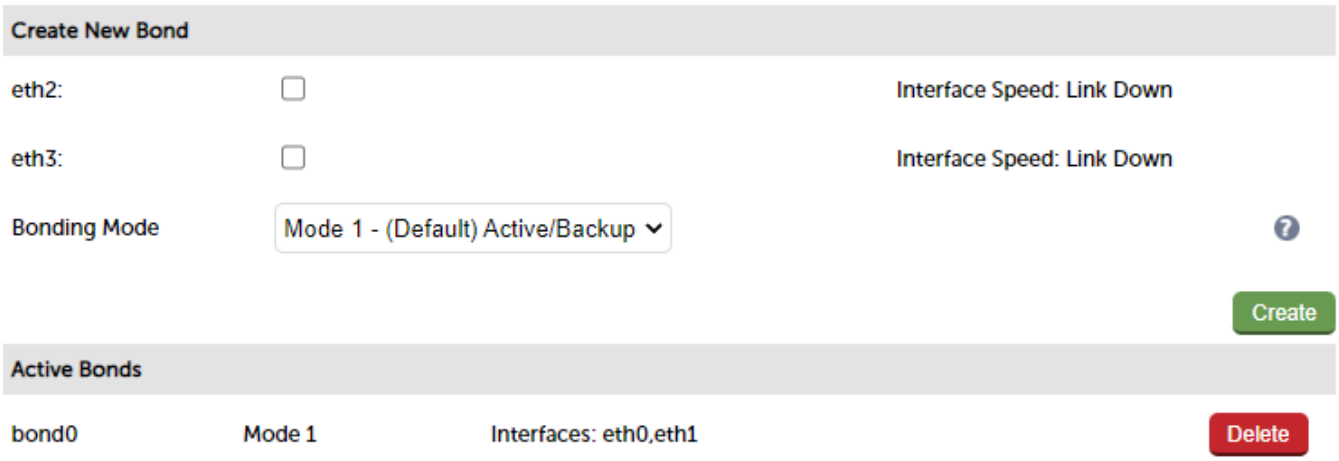
   Mode 0 - Balance round robin. Transmits packets in a numerical order from the first available Secondary through to the last.

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

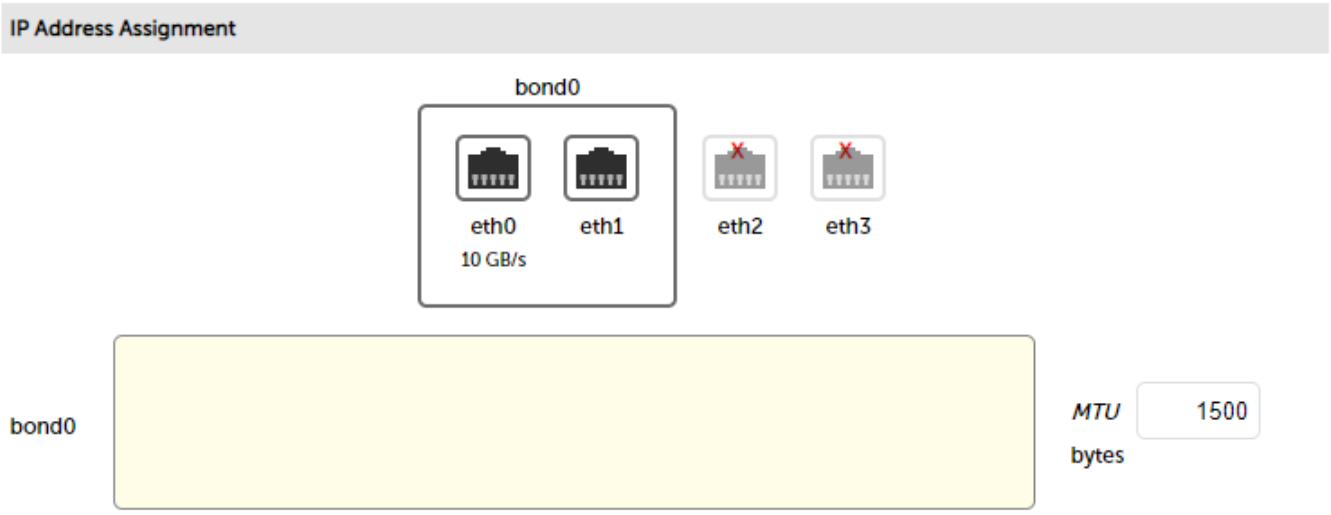
   Mode 4 - 802.3ad. Dynamic link aggregation mode. This mode requires a switch that supports IEEE 802.3ad.

Note	After changing the bonding mode a restart of the appliance is required for the setting to take effect.

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



Note	At this point the interfaces will still have the same IP settings configured previously. Once an IP address is defined for the bond and **Configure Interfaces** is clicked, these addresses will be removed and only the bond address will apply. If bonding is later disabled, these addresses will be re-applied to the interfaces.



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

Note	If you have a Primary and Secondary configured as an HA pair, make sure you configure bonding in the same way on both units. Failure to do this will result in heartbeat (Primary/Secondary communication) related issues.

Note	If your Real Servers, ESX hosts etc. support network bonding using Broadcom's SLB (Smart Load Balancing), this can cause issues in Layer 4 DR mode if older drivers are used. We have successfully tested SLB (Auto Fallback Disable) with driver version 15.2.0.5. Therefore at least this version is recommended.

Configuring VLANs

Native 802.1Q VLAN support can be enabled to load balance clusters on multiple VLANs.

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.

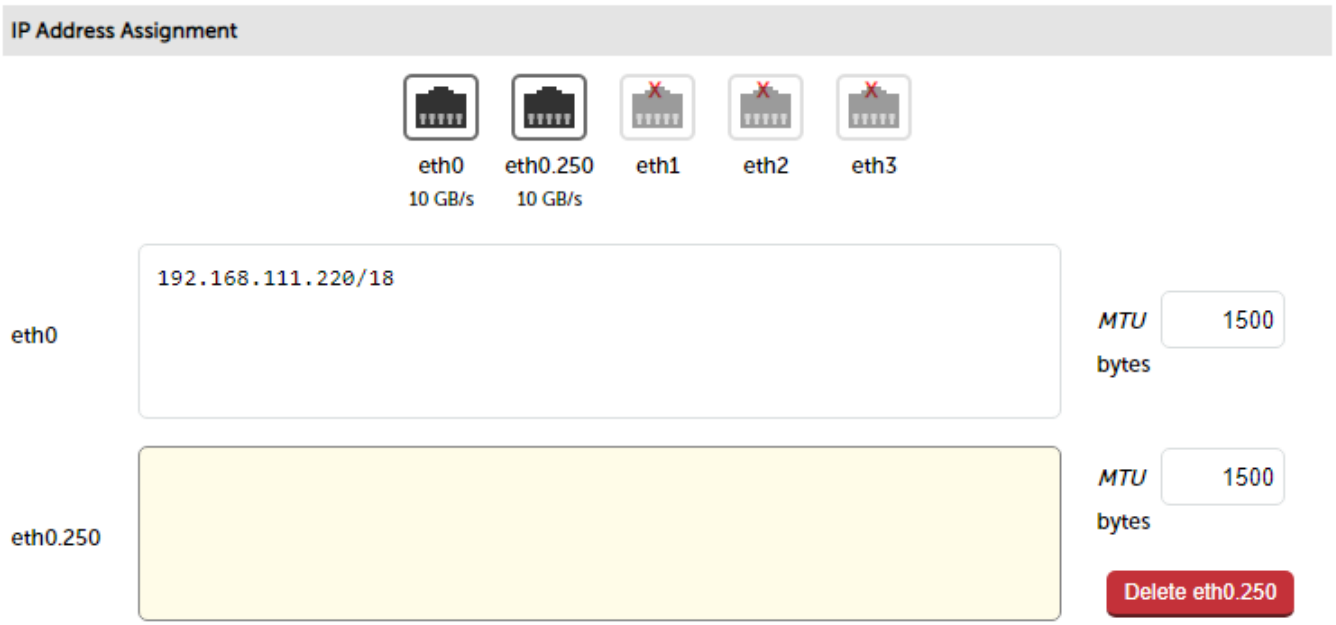
In ***trunk mode***, the switch passes on the raw VLAN frames, and the station must be configured to handle them. Trunk mode is usually used to connect two VLAN-carrying switches, or to connect a server or router to a switch.

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 will need to 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 interface (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 as shown below, the required IP address should be entered in this field.



6. Click **Configure Interfaces**.
7. To delete the VLAN definition, click the appropriate **Delete** button.

Note	If you have a clustered pair, don't forget to configure the same VLANs on the Secondary as these will not be replicated/created automatically.

NIC Offloading

NIC offloading is enabled by default. This will enable (where available) hardware NIC offloading.

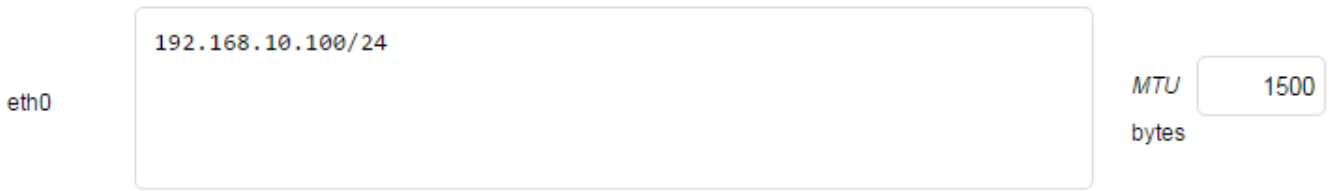
To Configure Offloading:

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 setting for an interface:

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



2. Enter the required MTU setting.
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. In the Default Gateway section define the default gateway as shown in the example below:

3. Click **Configure Routing**.

To configure Static Routes:

1. Using the WebUI, navigate to: *Local Configuration > Routing*.
2. In the Static Routes section configure the subnets & gateway addresses shown in the example below:

3. Click **Configure Routing**.

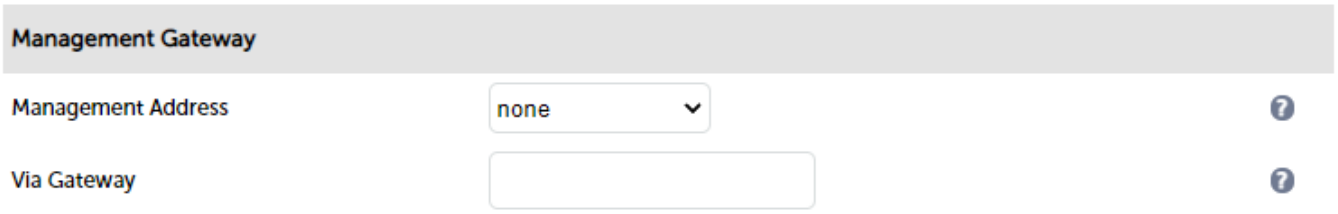
> Note
>
> Unlimited static routes can be defined, additional blank rows will be added to the WebUI screen as they're used.

Management Gateway

If required, a management address and associated gateway can be set. Once configured, traffic from the management address is routed via the management gateway rather than via the default gateway.

To configure the Management Gateway:

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



3. Select the required *Management Address* from the drop-down.

> **Note** The drop-down is populated with all the IP addresses that have been assigned to the various network interfaces.

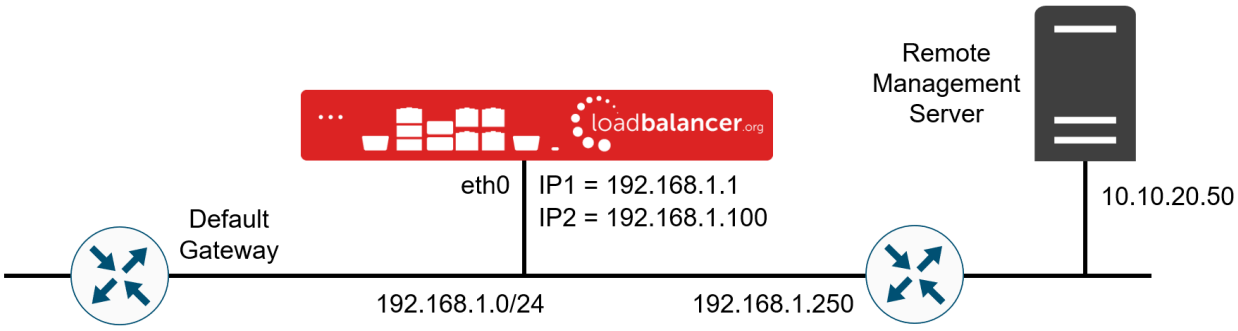
4. specify the required *Gateway* address.
5. Click **Update**.

> **Note** This is not a security feature, only a routing option. The WebUI is still accessible on all appliance IPs as before.

> **Note** Policy based routing is used to provide this feature. For more information, please refer to the PBR section below.

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:

> **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 Policy Based Routing. This can be configured in 2 ways:

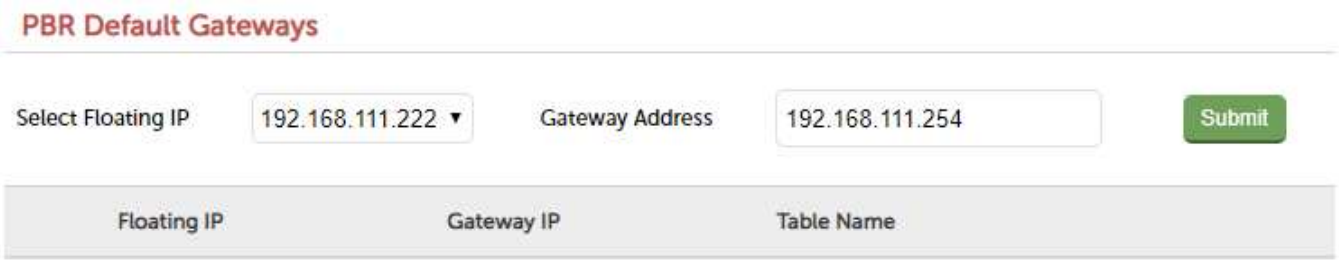
1. Using the WebUI (recommended).
2. At the command line via an SSH session or at the console (legacy).

Note	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, PBR can be used. Here, traffic can be routed based on the VIP that clients connect to rather than their source address.

1) Using the WebUI (Recommended)

To configure a VIP to return it's traffic via a custom gateway rather than via the default gateway using the WebUI:

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



2. Set the Floating IP to the required VIP address, e.g. **192.168.111.222**.
3. Set the Gateway Address to the required value, e.g. **192.168.111.254**.
4. Click **Submit**.
5. Once configured, the new default gateway will be displayed as shown below:

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

2) At the Command Line (Legacy)

To configure a VIP to return it's traffic via a custom gateway rather than via the default gateway at the command line:

At the console or via an SSH session, create a simple configuration file in **/etc/pbr.d/** with a .conf extension for that VIP. Files are expected to be called "<something>.conf", for simplicity we suggest using the VIP label so "VIP_NAME.conf".

Each config file must contain the desired gateway and the VIP IP address (this can be any LB IP!) in the following format:

```
GW="172.16.200.1"
VIP="172.16.200.37"
```

Optional options:

```
ROUTES= Adding this variable with the option "local" as below forces the script to only copy
the link local route for the VIP specified, not all link local routes.
```

```
FROM= Allows you to provide additional FROM rules, either a single address/subnet or multiple
addresses/subnets which would need to be space separated.
```

Examples:

```
ROUTES="local"
FROM="10.10.10.10/32 12.0.0.0/8"
```

Once you've created your .conf file, run the following command to start the PBR service:

```
service pbr start
```

Then run the following command to make it survive a reboot:

```
chkconfig pbr on
```

Important	PBR will also need to be configured on the Secondary appliance because the PBR config is not synchronized between Primary and Secondary. Either follow the same process used on the Primary or copy the config across using the following commands:

```
scp /etc/pbr.d/* root@lbslave:/etc/pbr.d/
ssh root@lbslave 'service pbr start'
ssh root@lbslave 'chkconfig pbr on'
```

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**.

It's also possible to control (start/stop) individual PBR configurations without affecting everything globally (avoiding a 'service pbr restart'). If you need to control an individual configuration and assuming your new PBR configuration is defined in file "INT_WEB_VIP.conf", issuing the following command would start that individual set of PBR rules:

```
service pbr start INT_WEB_VIP
```

Similarly, to stop those same rules the command below could be executed:

```
service pbr stop INT_WEB_VIP
```

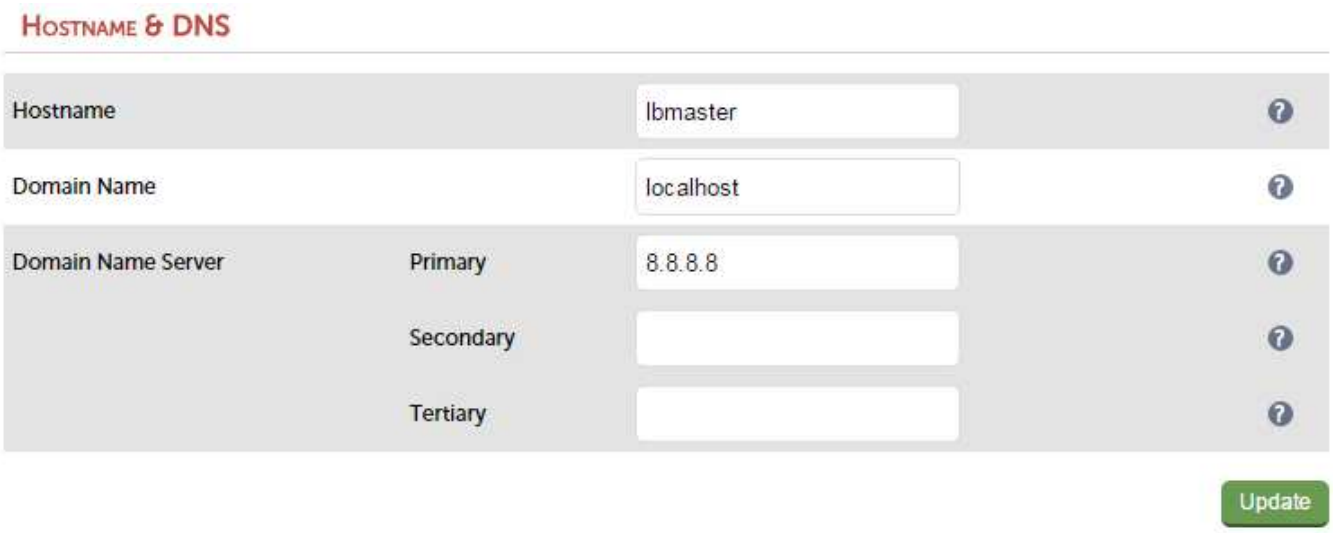
Important	The configuration file name should have the ".conf" removed from the parameter passed to the start/stop script.

Please contact support@loadbalancer.org if you need further assistance.

Configuring Hostname & DNS Configuration

To set the Hostname, Domain & DNS servers:

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



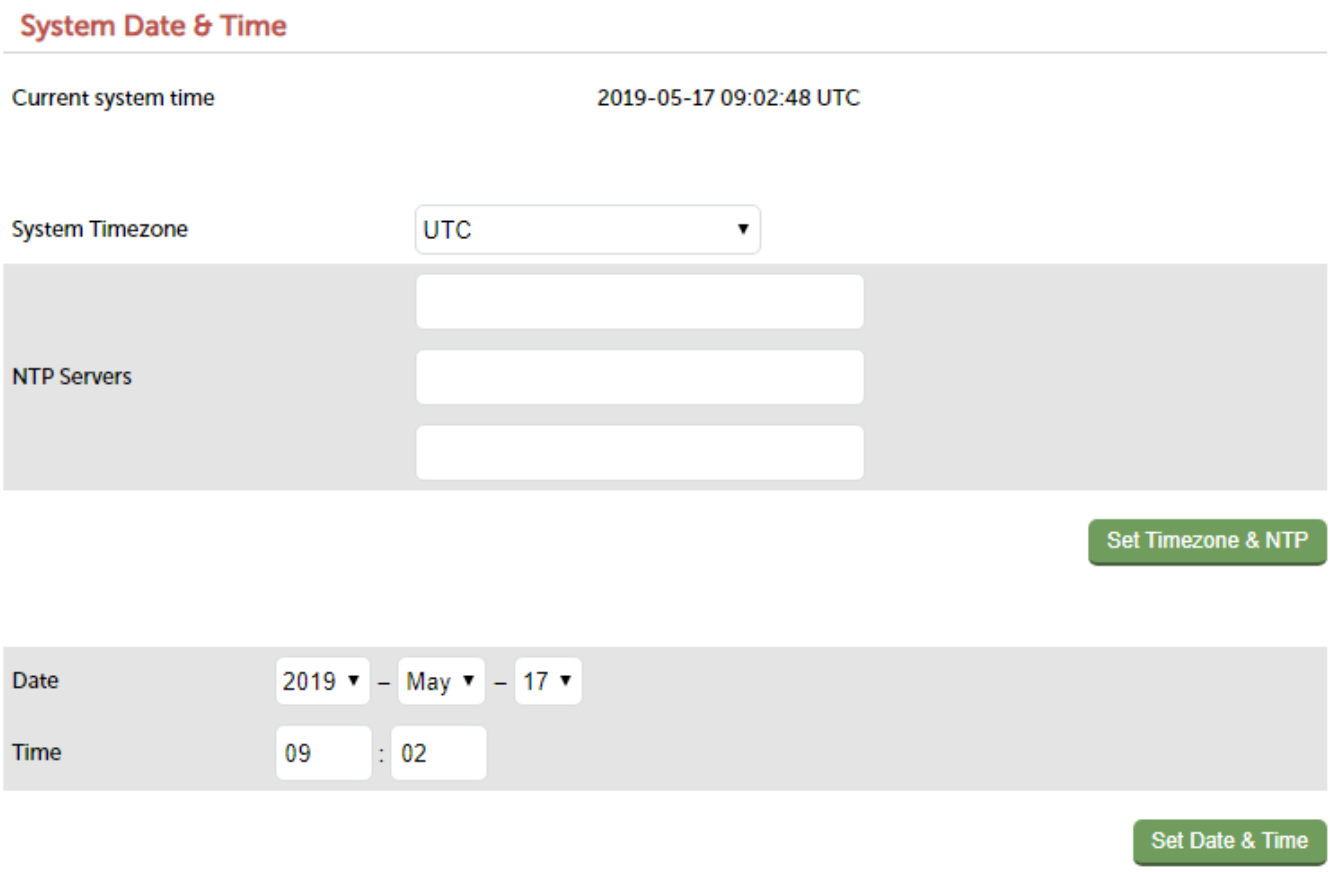
2. Specify the required *Hostname,* by default this is set to **lbmaster**.
3. Specify the Domain name, by default this is set to **localhost**.
4. Specify the required DNS server(s).
5. Click **Update**.

System Date & Time Configuration

Auto Configuration using NTP Servers

To configure NTP:

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



2. Select the required *System Timezone*.
3. Define your NTP servers 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**.

> Note
>
> When using a clustered pair (i.e. Primary & Secondary) date and time changes on the Primary will not be automatically replicated to the Secondary, therefore the date and time on the Secondary must also be set manually.

Appliance Internet Access via Proxy

The appliance supports the ability to access the Internet via a proxy server.

To set the Proxy Server's IP address & Port:

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

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

Note | For a clustered pair, this setting must also be manually configured on the Secondary.

SMTP Relay Configuration

The appliance can be configured with an SMTP smart host to receive all mail 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.

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

Note | For a clustered pair, this setting must also be manually configured on the Secondary.

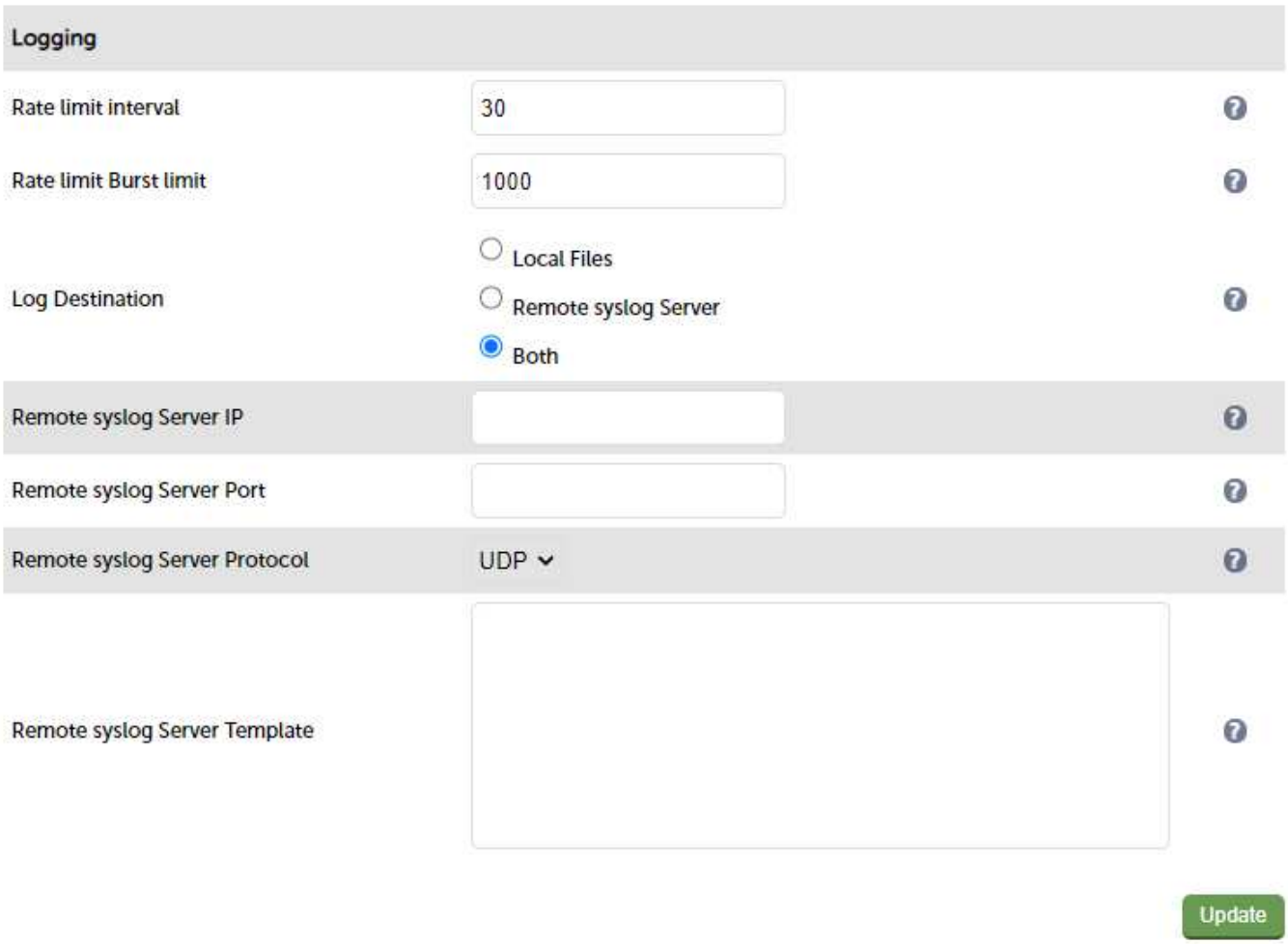
Syslog Server Configuration

The appliance supports the ability to write all logs either locally, to an external Syslog Server or both. The Syslog server may be specified by IP address or hostname.

To configure a Syslog server:

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

2. Scroll down to the *Logging* section.



3. Enter the required *Rate Limit Internal*, the default is 5 seconds.
4. Enter the required *Rate Limit Burst Limit*, the default is 200 messages.
5. Define whether logs should be written to *Local Files*, a *Remote Syslog Server* or *Both*.
6. If *Remote Syslog Server* or *Both* is selected, the following options also apply:

Option	Description
Remote Syslog Server IP	The server may be specified by IP address or hostname. If you use a hostname, make sure DNS is correctly configured on the load balancer.
Remote Syslog Server Port	Specify the Remote Syslog Server port.
Remote Syslog Server Protocol	Select the communications protocol, either TCP or UDP. 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 and this is the default.
Remote Syslog Server Template	Specify a Remote Syslog Server template (string format).

7. Click **Update**

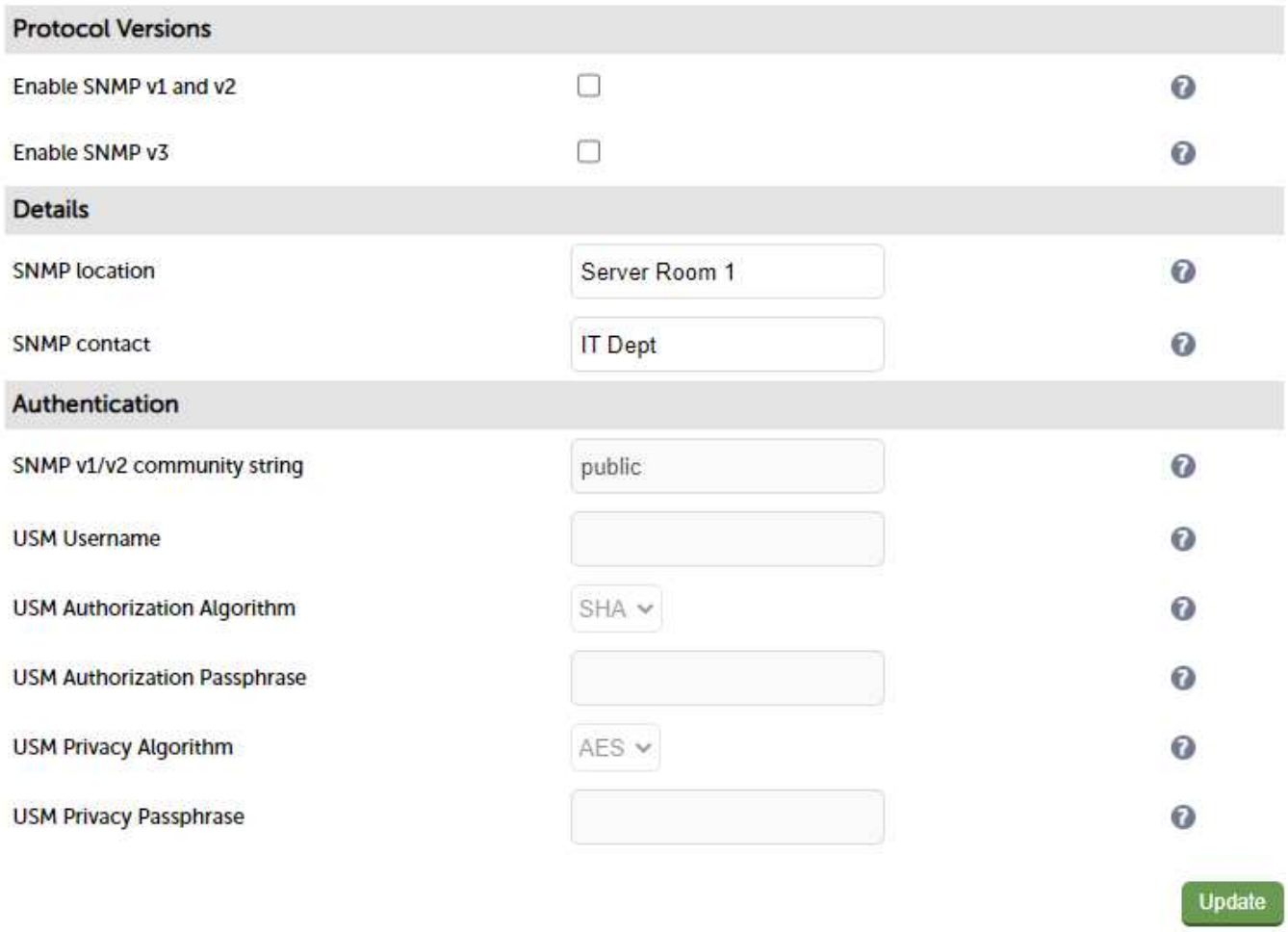
Note	For a clustered pair, this setting must also be manually configured on the Secondary appliance.

SNMP Configuration

The appliance supports SNMP v1, v2 and v3.

To Configure SNMP:

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



2. Using the checkboxes in the *Protocol Versions* section, enable the required SNMP version(s).
3. Enter a suitable *SNMP location* and *SNMP contact*.
4. For SNMP v1 & v2:
   - Enter a suitable *SNMP v1/v2 community string*.
5. For SNMP v3:
   - Enter a suitable *USM Username*, *USM Authorization Algorithm*, *USM Authorization Passphrase*, *USM Privacy Algorithm* and *USM Privacy Passphrase*.
6. Click **Update**.
7. Restart SNMPD using the **Restart SNMPD** button at the top of the screen.

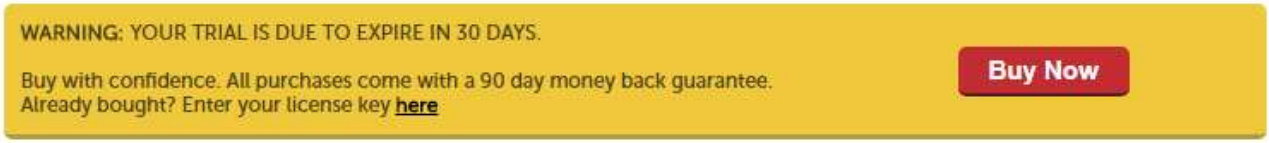
Note | For a clustered pair, this setting must also be manually configured on the Secondary appliance.

Note | More information about the various OIDs and associated MIBs for the appliance please refer to SNMP Reporting.

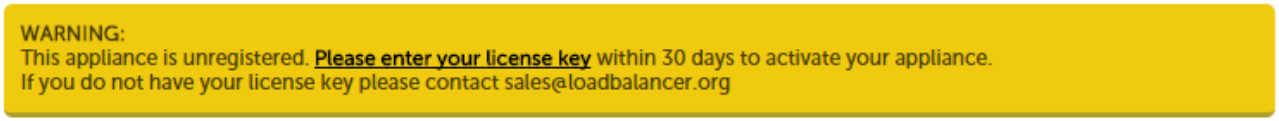
Installing License Keys

License keys are required for all appliances and must be applied within 30 days of power up as explained in the yellow information box in the WebUI.

The following message is displayed for a VA:

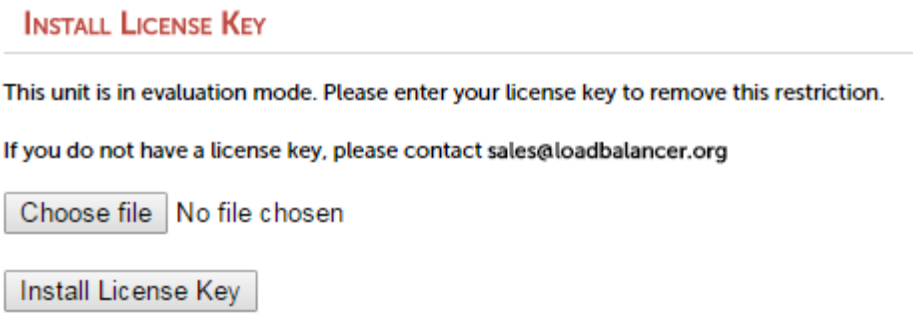


The following message is displayed for a hardware appliance:



To install the license key:

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



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

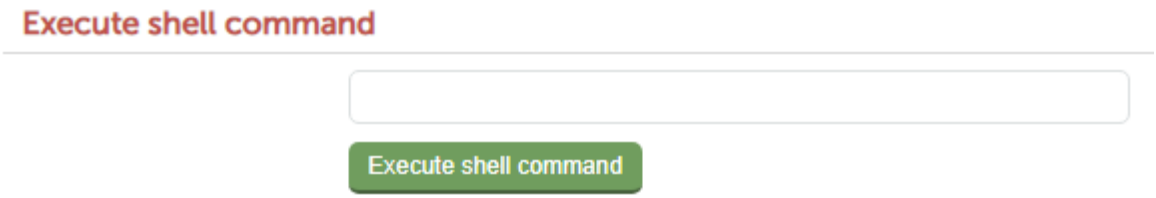
Note | Once the license is applied, these warning messages will no longer be displayed.

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*.



2. Enter the relevant command in the field.
3. Click **Execute Shell Command**.
4. The results of the command as well as any errors will be displayed at the top of the screen.

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

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**.

Restoring Manufacturer's Settings

The load balancer's settings can be reset to factory default values in two ways. In both cases this will remove all custom configuration from the load balancer. All VIPs, RIPs and other settings will be removed and the IP address configured for eth0 will be set to 192.168.2.21/24.

Using the WebUI

To restore settings:

1. Using the WebUI, navigate to: *Maintenance > Backup & Restore > Restore Tab*.
2. Click **Restore Manufacturer's Defaults**.

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

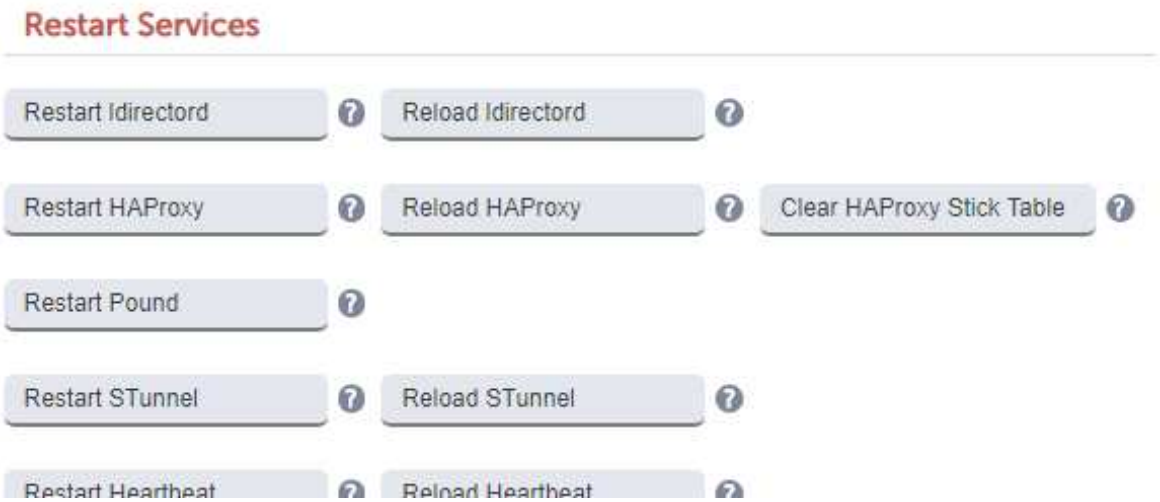
'root' user console and SSH password access are disabled by default. These options can be enabled using the WebUI option: *Local Configuration* > *Security*. You'll need to Set *Appliance Security Mode* to **Custom**, enable the required option(s) and click **Update**.

Restarting & Reloading Services

The various services running on the appliance can be manually reloaded or restarted if required. This is normally only required for HAProxy, Pound, STunnel and Heartbeat when configuration changes are made.

To restart / reload services:

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



2. Click the relevant restart or reload button.
3. Click **OK** 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, reload the configuration. If you are using stick tables for persistence the entries will be copied between processes. HAProxy will start a new process (leaving the old one) with the new configuration. New connections will be passed onto this process, the old process will maintain existing connections and eventually terminate when there are no more connections accessing it.

Note	If you have long lasting TCP connections it can take quite some time for this old process to terminate, leaving those users running the old configuration. If this is taking too long - See Restart HAProxy.

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 SSL termination services during the restart.

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

Reload STunnel
Restarting STunnel may result in a loss of 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 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 load in any changes made to the configuration file.

Restart Collectd
This will not clear the previously collected data. Note that collectd will not start if graphing of all services is disabled.

Restart SNMPD
Restart the SNMP 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 it's 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 are 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.

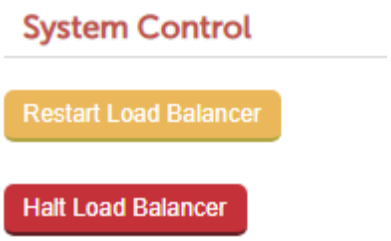
Reload GSLB
Reload GSLB services to make live any changes to configuration. This should not impact live services.

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*



2. Select the required option:

   Restart Load Balancer - *Shutdown and restart the appliance*

   Halt Load Balancer - *Shutdown and halt the appliance*

Appliance Software Updates

Loadbalancer.org continually develop and add new and improved features to the appliance. To ensure that customers can benefit from this and can also receive bug fixes and security updates, Loadbalancer.org support both online and offline updates to ensure that all customers who have a valid maintenance and support contract are able to update their appliances. A security updates only option is also available for customers that don't require the benefits of our complete support package.

Note	Since services may be restarted during the update process we always recommend performing the update during a maintenance window. For some updates a full appliance restart is required. In these cases a restart notification message will be displayed after the update is complete.

Checking the Current Software Version

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

Online Update

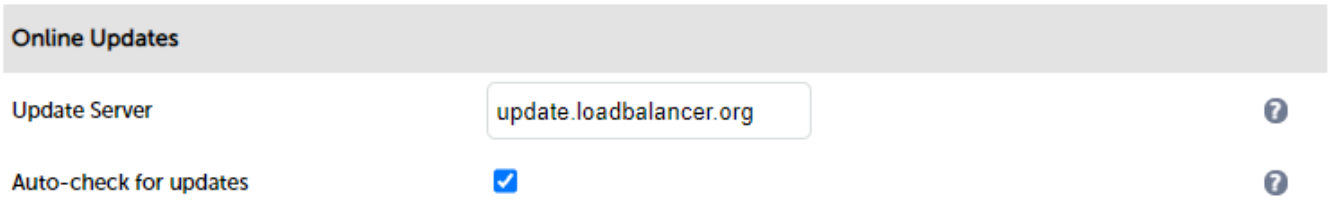
Auto-Check for Updates

For v8.6.0 and later, the appliance periodically contacts the Loadbalancer.org update server (update.loadbalancer.org) and checks for updates. This is the default behavior and can be disabled if preferred. If an update is found, a message will be displayed at the top of the screen as shown below:

The user can then initiate the update process using the **Online Update** button.

To configure Online Updates:

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



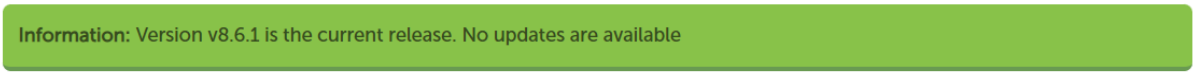
3. Configure the required settings.
4. Click **Update**.

Manual Check for Updates

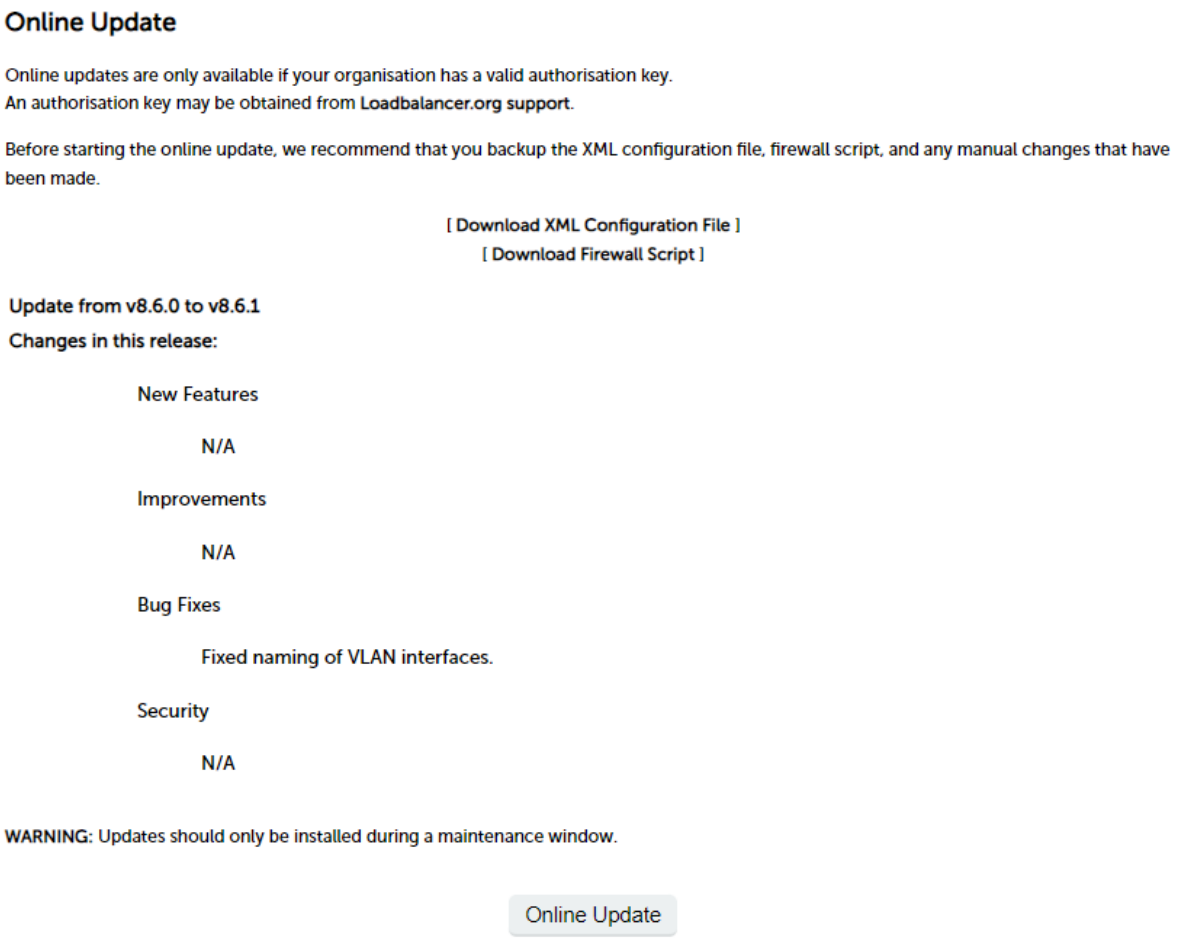
If automatic updates are disabled, the update check must be initiated manually.

To initiate a manual online update check:

1. Using the WebUI, navigate to: *Maintenance > Software Update*.
2. Select **Online Update**.
3. If the latest version is already installed, the following message will be displayed:



4. If an update is available, information similar to the following will be displayed - this shows a v8.6.0 to v8.6.1 update:



5. Click the **Online Update** button to start the update process.

Note | Do not navigate away whilst the update is ongoing, this may cause the update to fail.

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

2. If there are any specific post upgrade requirements such as a service restart these will be displayed on the screen after the installation completes.

Notes

1. As indicated in the WebUI, we recommend that you should backup your XML configuration file and any other configuration that has changed from default settings before running the update. This can be done using the WebUI backup options under: *Maintenance > Backup & Restore > Backup*.
2. Make sure that the load balancer is able to access the Internet - if you have a proxy server, this can be defined using the WebUI option: *Local Configuration > Physical Advanced Configuration* and configuring the *Network Proxy* section.
3. Make sure that the default gateway is set correctly (*Local Configuration > Routing*).

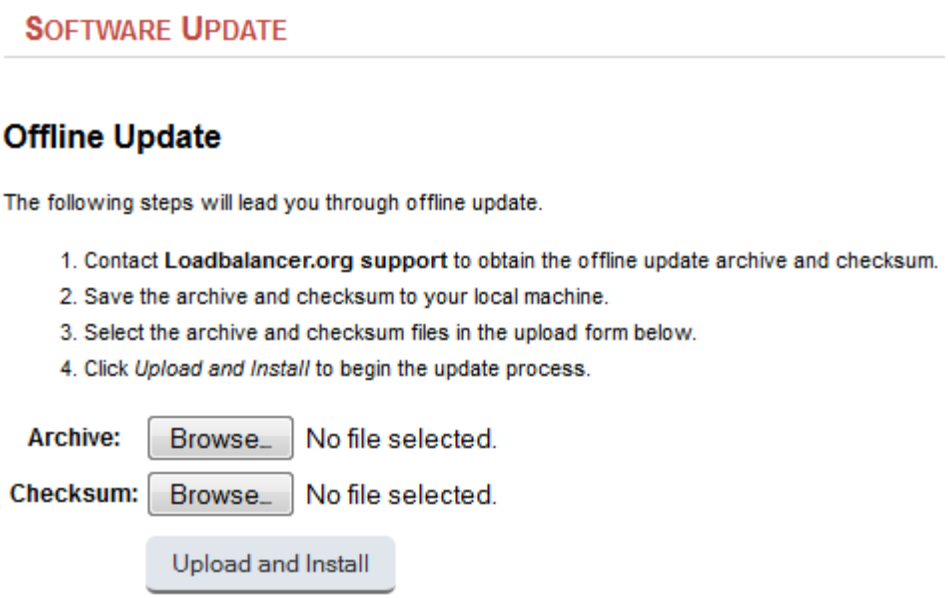
4. Make sure that a valid DNS server is specified (*Local Configuration > Hostname & DNS*).

Offline Update

If the load balancer does not have access to the Internet, Offline Update can be used.

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:



4. As explained in the on-screen text, contact Loadbalancer.org support to obtain the archive & checksum files.
5. Browse to and select these files.
6. Click **Upload and Install**.

Updating a Clustered Pair

Note	Since services may need to be 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 first. The updates are incremental, so we recommend installing each update in turn, ignoring calls to restart services or reboot the appliance until all available updates have been installed and the appliance is fully up to date.
2. Next, restart services or reboot the appliance as directed.
3. Now update the Primary unit in the same way.

Note

For a clustered pair, we strongly recommend fully testing & validating the Primary/Secondary failover process before going live. If testing was not carried out before go-live, we recommend scheduling a maintenance window to do this to ensure that everything is configured correctly and you are aware of the process. For detailed testing & verification steps please refer to Testing & Verifying Primary/Secondary Replication & Failover.

Appliance Security Features

The appliance includes a number 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, 3 security modes are provided:

- **Secure - (default)** - in this mode:
  - the WebUI is accessible on HTTPS port **9443**. If you attempt to access the WebUI on HTTP port **9080** you will be redirected to HTTPS port **9443**
  - access to the *Local Configuration > Execute shell Command* menu option is disabled
  - the ability to edit the firewall script & the lockdown wizard is disabled
  - 'root' user console & SSH password access are disabled
- **Custom** - in this mode the following check-box options can be configured to suit your requirements:
  - *Disable Console Access* - prevent 'root' access via the console
  - *Disable SSH Password Access* - prevent 'root' access via SSH
  - *Web User Interface via HTTPS only* - control whether the WebUI is only accessible on HTTPS port **9443** only or via both HTTPS port **9443** and HTTP port **9080**

    Note

    When **Custom** is selected access to the *Local Configuration > Execute shell Command* menu option is automatically enabled.
- **Secure - Permanent** - this mode is the same as **Secure** but the change is ***irreversible***

Important

Only set the security mode to **Secure - Permanent** if you are 100% sure this is what you want!

To configure the Appliance Security Mode & related options:

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

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 WebUI, the default is **9443**.
5. Select the required SSL certificate for the WebUI. Certificates can be created/uploaded using the WebUI menu option: *Cluster Configuration > SSL Certificate*. If no certificates are available, the appliance's default self-signed certificate will be used.
6. Specify the required cipher, 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
```

Users & Passwords

Linux 'root' User Account

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

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**.

The root password can be changed at the console, or via an SSH session using the following command:

```
# passwd
```

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 defined during instance deployment, run the following command:

```
$ sudo su
```

WebUI User Accounts

By default the appliance includes four predefined WebUI user accounts. The default usernames, passwords, group

membership and their primary use are:

Username	**Default Password**	**Default Group**	**Description** (please also refer to the group table below)
configuser	configuser	config	appliance administration account
loadbalancer	(configured during the Network Setup Wizard)	config (*)	appliance administration account
reportuser	reportuser	report	viewing the appliance configuration, reports & logs
maintuser	maintuser	maint	same as reportuser plus can also take servers on/off line & create the support download archive file

(*) It's not possible to change the default group for the 'loadbalancer' user account.

Note

These are Apache .htaccess style accounts and are not related to the local Linux OS level accounts.

The permissions for each group are shown below:

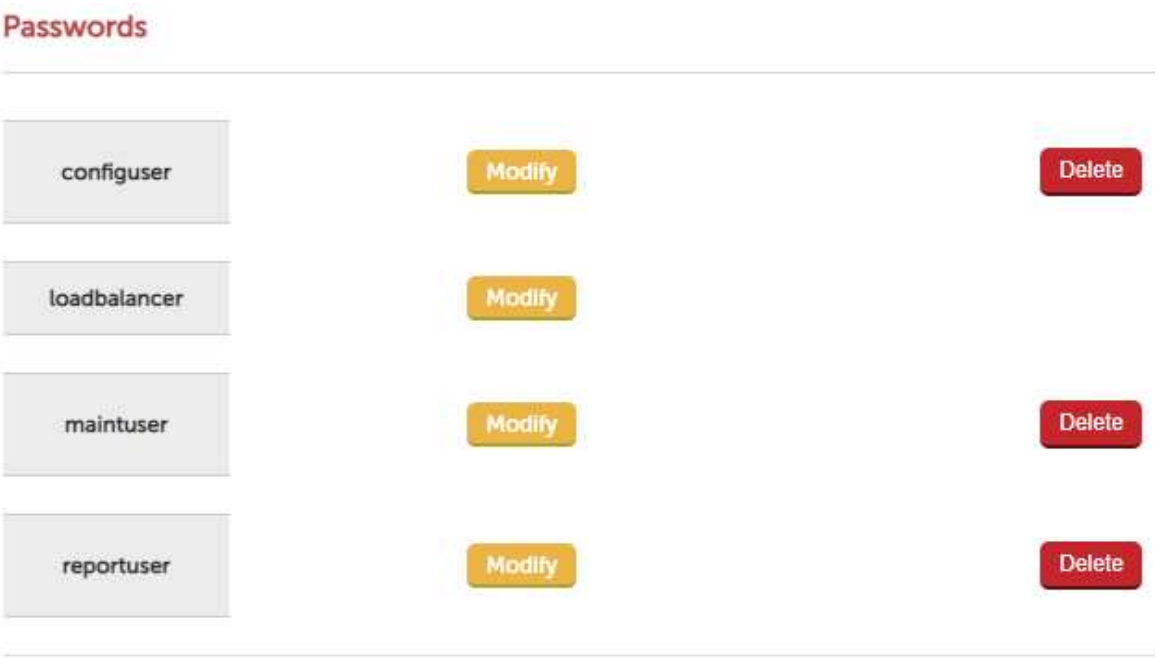
	Menu/Permissions							
Group	System Overview	Local configuration	Cluster Configuration	Maintenance	View Configuration	Reports	Logs	Support
config	Full	Full	Full	Full	View	Full	View	Full
report	View	None	None	None	View	Full	View	View
maint	Full	None	None	None	View	Full	View	Full

It's also possible to define users who will be authenticated by an external LDAP/ADAuth system as described in the *Adding New Users* section below.

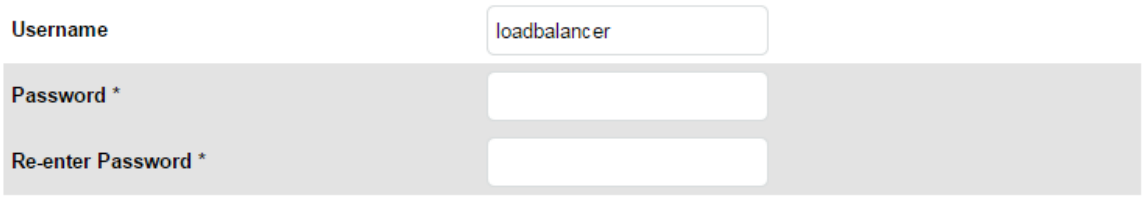
Modifying User Passwords

To modify a user's password:

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



2. Click the **Modify** button next to the relevant user.
3. Now change the password for the selected user:



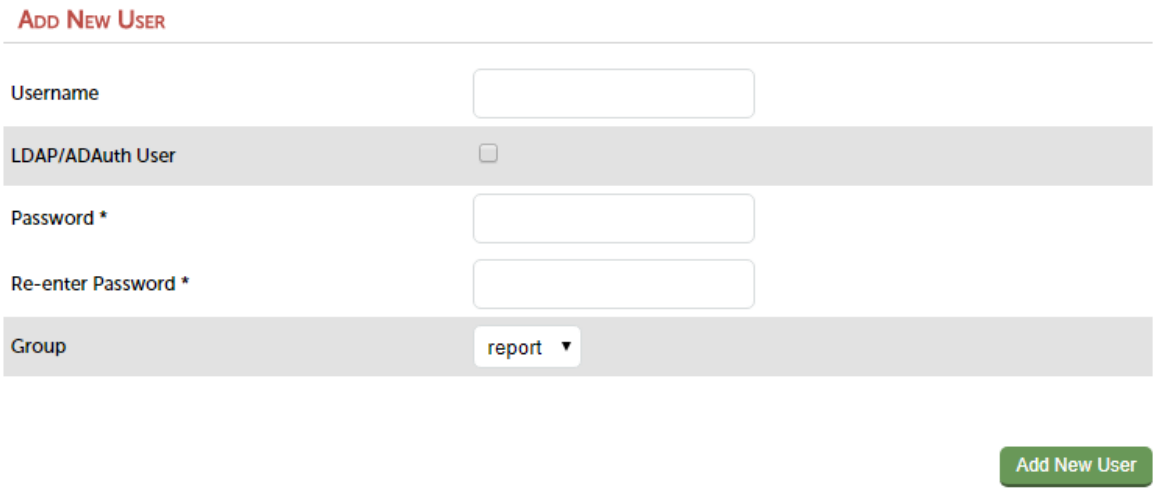
Note | Passwords cannot contain the double quotation mark (").

4. Click **Edit User**.

Adding New Users

To add new users:

1. Using the WebUI, navigate to: *Maintenance > Passwords*.
2. Use the following section:



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 the *External Authentication* section below.

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

> Note | Passwords cannot contain the double quotation mark (").

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 you'll need to login as root to the console/SSH session. The *htpasswd* command can then be used as shown below:

```
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 option: *Local Configuration* > *Security*. You'll need to Set *Appliance Security Mode* to **Custom**, enable 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 simply enter their credentials for that system to access the appliance. It's important to remember that the username defined in the **Add New User** screen must be the exact same username defined in the external authentication system.

To demonstrate this feature, an Active Directory example is presented below, the steps required to configure against other external authentication systems are similar.

Configuring External AD Authentication

To configure external authentication, either at the console or using an SSH session, login as root 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 option: *Local Configuration* > *Security*. You'll need to Set *Appliance Security Mode* to **Custom**, enable the required option(s) and click **Update**.

Note	Do not run lbauthconfig from the WebUI.

The following will be displayed:

```
[root@lbmaster ~]# 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 Active Directory, enter **4** and hit <ENTER>

```
Step 2. Enter the hostname/address and port of the authentication server.
Hostname/address: 192.168.112.1
Port [1-65535 (3269)]:
```

>> Enter the hostname or IP address & port (default is **3269**) of your domain controller, e.g. **192.168.112.1** 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
```

>> Enter 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 [userPrincipalName]:
```

>> Enter the LDAP attribute to authenticate against (default is **UserPrincipalName**) and hit <ENTER>

```
Step 5. Enter the credentials for a user who can browse the directory.
Username: lbuser
Password:
```

>> Enter the credentials for a user who can browse AD, e.g. **lbuser** and hit <ENTER>

```
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:
```

>> Enter the username & password of the first AD authenticated user, e.g. **tom@lbtestdom.com** and hit <ENTER>

The new user will be added as shown below:

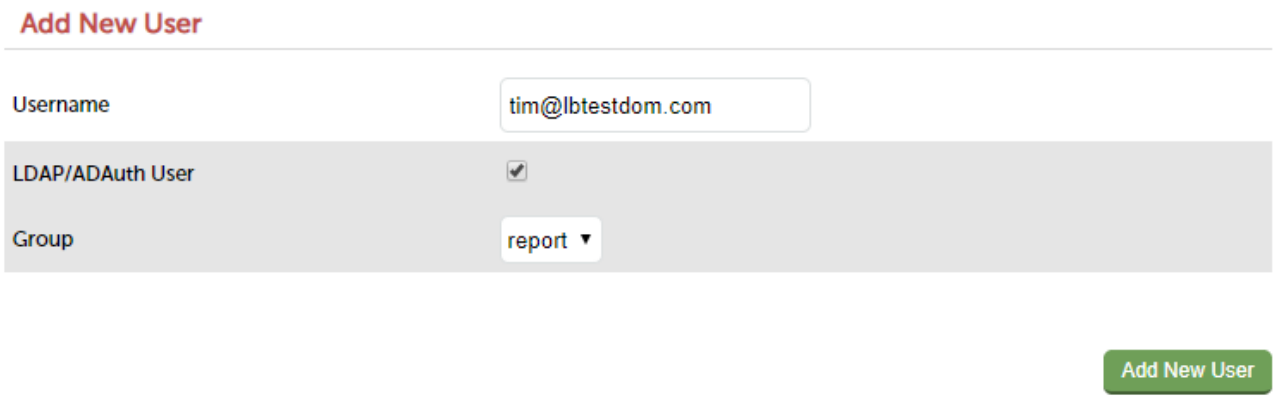
```
Test authentication succeeded.
Creating backup of /var/www/html/lbadmin/.htaccess at /etc/loadbalancer.org/bkup/.htaccess-2019-12-12T15:25:58.621399
Creating backup of /etc/loadbalancer.org/groups at /etc/loadbalancer.org/bkup/groups-2019-12-12T15:25:58.621399
Creating backup of /etc/loadbalancer.org/passwords at /etc/loadbalancer.org/bkup/passwords-2019-12-12T15:25:58.621399
Writing new /var/www/html/lbadmin/.htaccess File.
Writing new /etc/loadbalancer.org/groups File.
Adding tom@lbtestdom.com to /etc/loadbalancer.org/groups File.
Adding password for user tom@lbtestdom.com

Finished.
[root@lbmaster ~]#
```

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 AD 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:



1. Specify the same username as the AD / RADIUS user to be added, e.g. **tim@lbtestdom.com**.
2. Enable (check) the *LDAP/ADAuth User* checkbox.
3. Select the required 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 AD credentials.

Firewall Configuration

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.

If you want to configure firewall rules, here are some points to consider:

- All Virtual Service connections are dealt with on the INPUT chain not the FORWARD chain
- The WebUI runs on HTTP port 9080 (disabled by default) and HTTPS port 9443
- SSH on the load balancer listens on the standard port (22)
- SNAT & DNAT is handled automatically for all layer 4 NAT mode (LVS) and layer 7 (HAProxy) based Virtual/Real load balanced services
- You can use the standard Linux filters against spoofing attacks and syn floods
- LVS has built in DOS attack filters that can be implemented

Note

Plenty of extra information is available on the Internet relating to Linux Netfilter and LVS, if you need any assistance please contact support@loadbalancer.org.

Manual Firewall Configuration

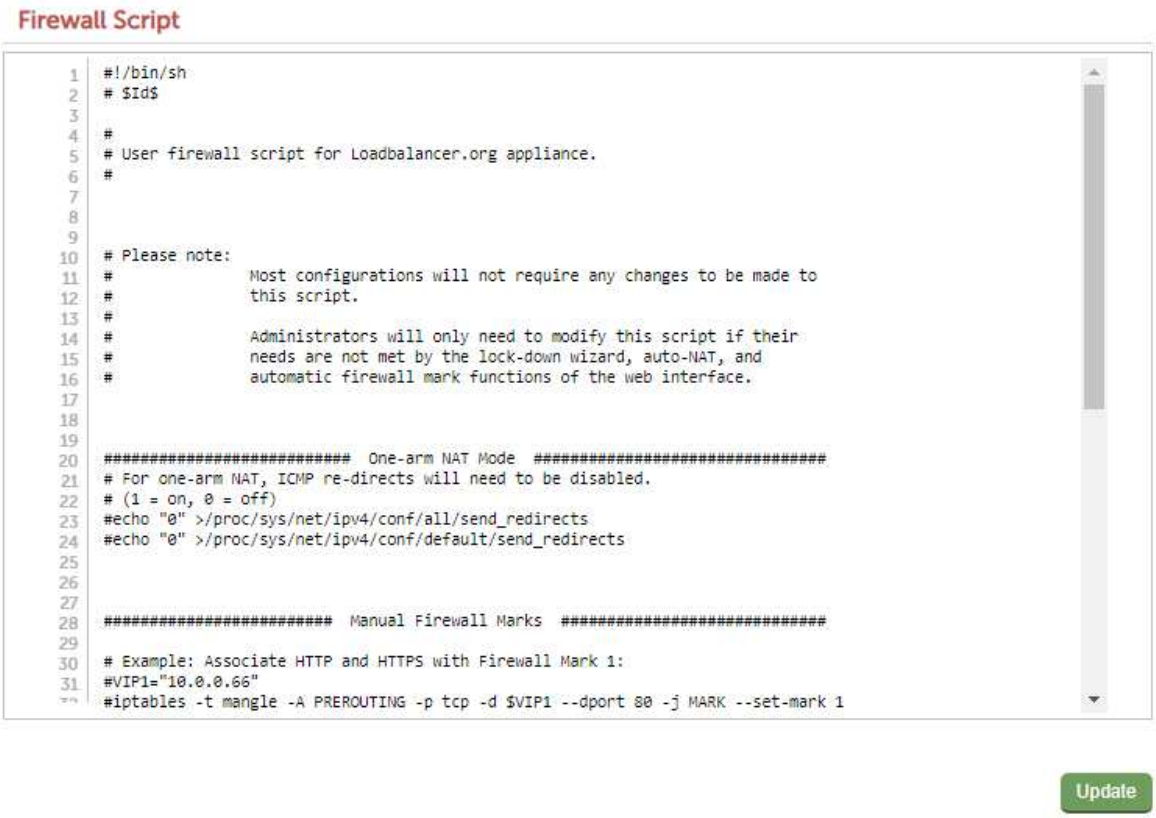
The firewall can be configured manually using the WebUI based script editor. This enables iptables rules and any other required commands to be easily defined. The form allows you to directly edit /etc/rc.d/rc.firewall.

Custom rules can be configured, or for belt & braces security your external firewall settings can be replicated on to the load balancer for multi-layer security.

If you're planning to use NAT mode you may want to use the load balancer as your main firewall but we recommend that it is better and simpler to keep your firewall separate from the load balancer, especially if you want to set up VPNs etc. You can also use the firewall script to group ports together using Firewall Marks. For more information please refer to Firewall Marks.

To configure custom firewall rules:

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



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

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

4. Click **Update**.

Note	For a clustered pair, firewall script changes must also be manually configured on the Secondary.
Note	Be careful !! Make a backup before changing this script so that you know you can roll everything back if you cause a problem. A backup can be created using the WebUI menu option: *Maintenance > Backup & Restore > Make Local Firewall Script Backup*.

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 that is written automatically when **Update firewall lock down** is clicked. The file depends on the rc.lockdownwizard script and the load balancer's 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 rules. The WebUI writes the definitions rc.lockdownwizard.conf. You can modify rc.lockdownwizard via ssh or from the WebUI using the **Modify the firewall lock down wizard script** button. Apart from this link there is no other influence from the WebUI.

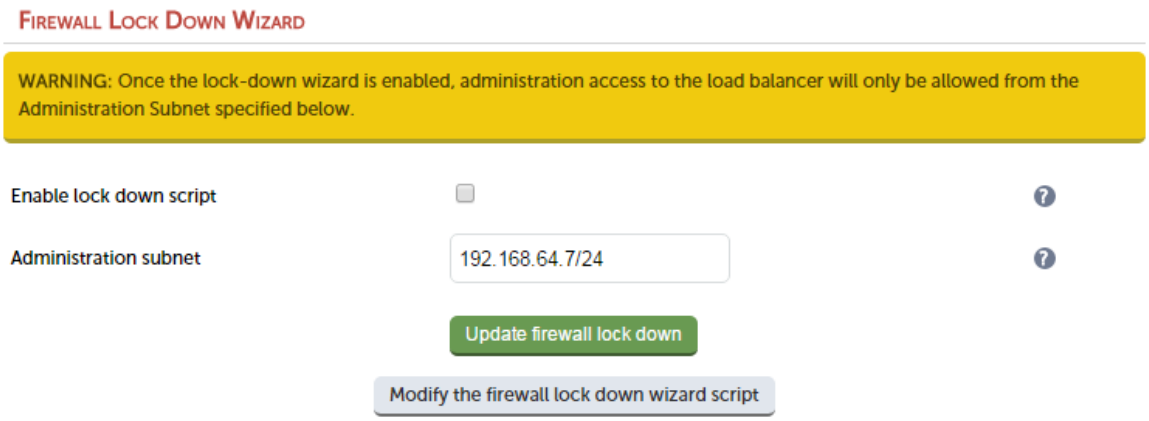
Note

'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**.

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, 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:

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



3. Define your administration subnet/host in the *Administration subnet* field.

Note

Make sure that the subnet mask is correct - by default a /24 mask is used. To lock down access to a single host use <IP address>/32, e.g. 192.168.2.1/32.

4. Click **Update firewall lock down**.

Note

For a clustered pair, the lockdown wizard must be run on each appliance.

To disable the lock-down script:

1. 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 clear the firewall tables completely use the following command at the console: /etc/rc.d/rc.flush-iptables.

Conntrack Table Size

By default the connection tracking table size is set to 524288 and is fine 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:

```
ip_conntrack: table full, dropping packet.
```

To modify this setting:

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

Note

For a clustered pair, this setting must also be manually configured on the Secondary.

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 appliance's 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 option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, enable the required option(s) and click **Update**.

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

```
[root@lbmaster ~]# lbsecure


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:  192.168.64.0/18

Working...
   Generating new SSH keys...
   SSH keys replaced.

   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.

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 command *ibinsecure* can be used.

For a clustered pair, run *lbinsecure* on both Primary and Secondary to completely reverse the configuration applied by running lbsecure.

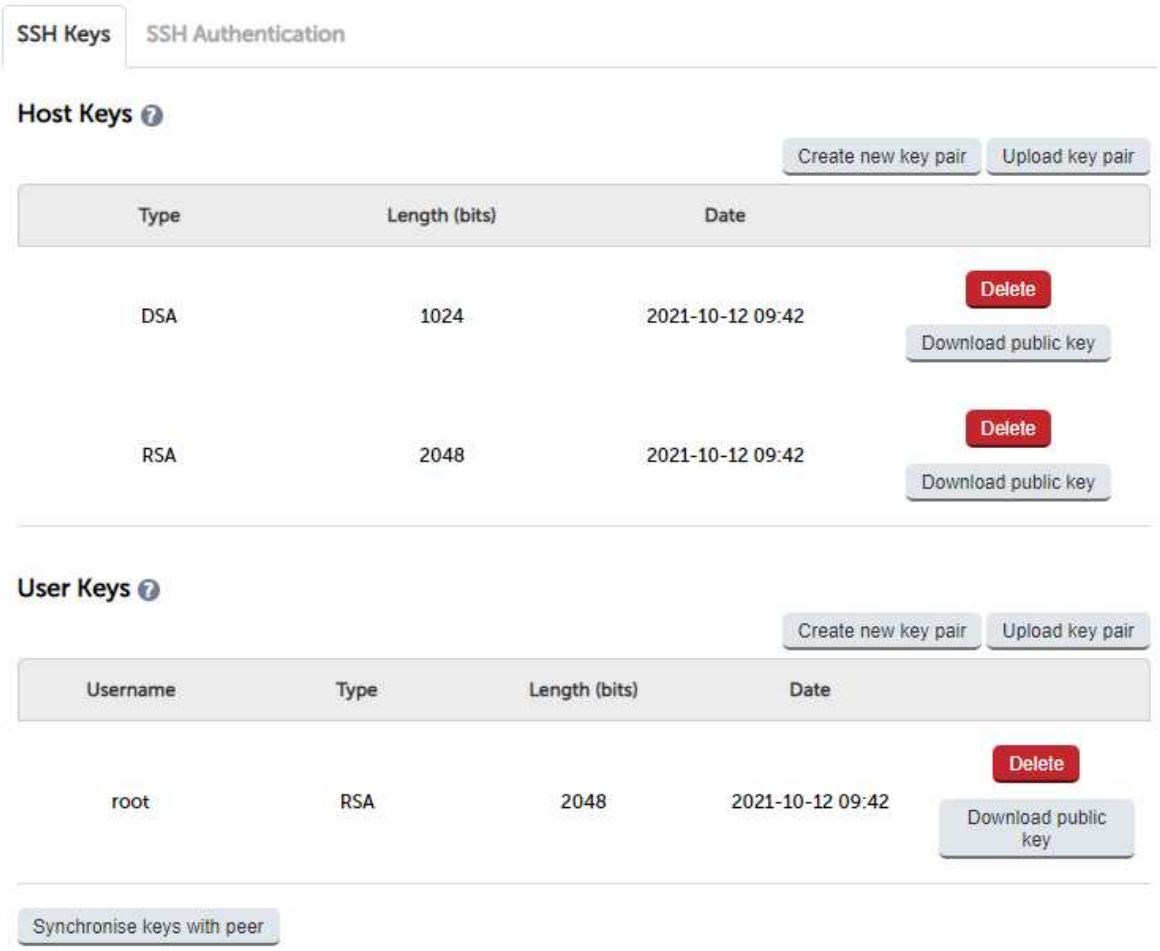
SSH Keys

This menu option enables SSH keys to be managed.

Note | Since SSH keys are managed by the appliance and under normal circumstances do not require user intervention, this menu option will not normally be used.

To view/manage SSH keys:

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



- 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 known key(s) of hosts that have been previously connected to or have been preconfigured. In an HA pair you you will see the peer appliance keys.
  - **User Keys (authorized_keys)** - the public key(s) of remote hosts that can log in as the specified user. In an HA pair you you will see the peer appliance keys.

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*

Configuration	File & Location
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

Chapter 6 - Configuring Load Balanced Services

Introduction

As discussed here, 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.

Layer 4 persistence is based on source IP address by default. The time out value is in seconds and each time the client makes a connection the timer is reset, so even 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. Check *View Configuration > Network Configuration* to ensure that the floating IP address has been activated correctly. They will show up as secondary addresses/aliases.

Multiple ports can be defined per VIP, 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 (VIPs)

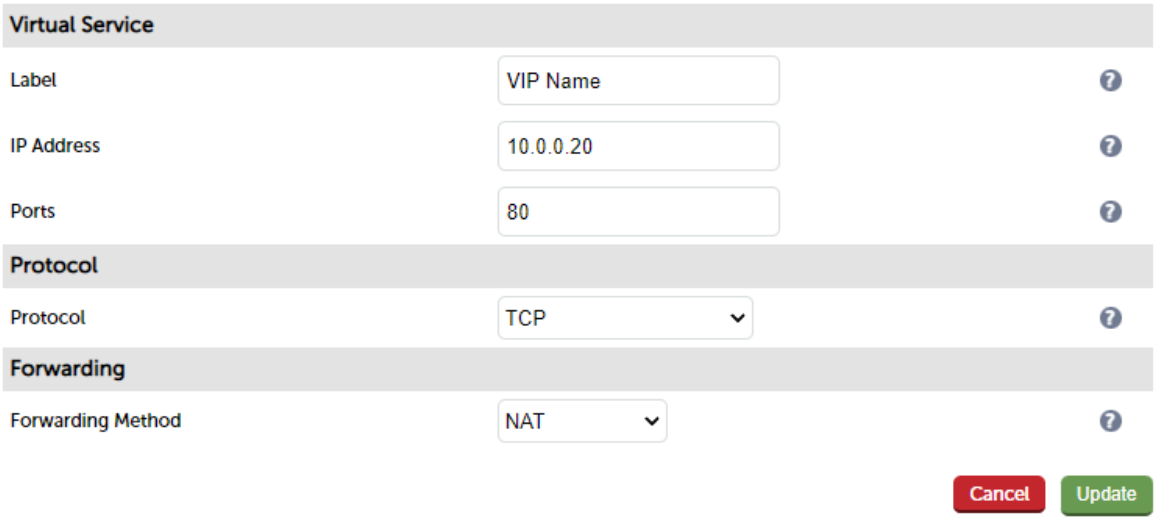
Virtual services can be created in 2 ways, either by defining a new VIP from scratch where the required settings must be defined manually, or by using the duplicate VIP feature.

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'd create a VIP for a web cluster, another for an FTP cluster and a third for a SIP cluster. Multiple ports can also be specified for each VIP.

Defining 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**.



3. Enter an appropriate *Label* (name) for the new Virtual Service.
4. Enter the required IP address in the *Virtual Service IP address* field.
5. Enter the required port(s) in the *Virtual Service Ports* field, separate multiple ports with commas, specify a range with a hyphen and specify all ports using an asterisk (*).

> Note
>
> Several ports are used by the appliance and therefore cannot be used for Virtual Services. For full details please refer to Ports Used by the Appliance.

6. 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
7. 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 looses 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. Now proceed to define the RIPs (Real Servers) as described here.

Duplicating an Existing Layer 4 VIP

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

> **Note** This option will copy all Virtual Service settings along with all associated Real Servers. After duplicating, you'll need to change either the IP address or port. If this is not done, the new VIP will clash with the original VIP and will not load. All other settings can remain the same if required.

To duplicate an existing layer 4 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 4 VIP

When first adding a Virtual Service, only certain settings can be configured, others are set at their default value to simplify initial configuration. These values can be changed after the Virtual Service has been created by clicking **Modify** next to the relevant Virtual Service. Additional settings that can be changed are:

Section	Setting	Description
Connection Distribution Method	Balance Mode	Select the required method to distibute new connections. The options are: • **Weighted Least-Connection** - assign more jobs to servers with fewer jobs, relative to the Real Server's weight (the default). • **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. Note: When using this mode, 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.
Persistence	Enable	Enable (the default) or disable persistence. 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. Note: If *Protocol* for the Virtual Service is set to 'One Packet Scheduling', persistence will be based on SIP Call-ID. Note: If your Real Servers cannot keep session state persistence themselves, then you will obtain performance but not reliability benefits from a load balancer.
	Timeout	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). Persistence will last for ever if the client clicks on a link within that period.

Section	Setting	Description
	Granularity	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	Check Type	Specify the type of health check to be performed on the Real Servers. As the Check Type drop-down is changed, the related field list changes. The options are: • **Negotiate** - Scan the page specified in *Request to Send*, and check the returned data for the *Response Expected* 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. Note: For full details of all layer 4 health check options, please refer to Health Checks for Layer 4 Services.
	Check Port	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.

Section	Setting	Description
Feedback	Feedback Method	The method the load balancer uses to measure to performance of the Real Servers. The options are: • **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). 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 ((92/10) * requested_weight) to find the new weight. Using this method an idle Real Server will get 10 times as many new connections as an overloaded server.
Fallback Server	IP Address	The server to route to if all of the Real Servers in the group fail the health check. The local nginx fallback server is configured for the ports 80 and 9081 (configured to always show the index.html page). When using HAProxy Layer 7 the nginx server port 80 is automatically disabled. You can also configure the fallback server to be a 'Hot Spare' if required. For example you have one server in the cluster and one fallback they will act as a Primary/Secondary pair.
	Port	Set the fallback server port, for DR mode leave this blank as it must be the same as the VIP.
	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.

Note

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

Creating Layer 4 Real Servers (RIPs)

You can add an unlimited number of Real Servers to each Virtual Service. In DR mode, since port redirection is not possible the Real Server port field is not available and the port is automatically set to be the same as the Virtual Service, whilst for a NAT mode Real Server, it's possible to configure the port to be the same or different to the Virtual Service's port.

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.

3. Enter an appropriate *Label* (name) for the new Real Server.
4. Enter the required IP address in the *Real Server IP Address* field.
5. Enter the required port in the *Real Server Port* field. This only applies to NAT mode, in DR mode port redirection is not possible so by default the port is the same as defined in the VIP.
6. 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.
7. 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.
8. If Minimum Connections is set with other values, the server will receive new connections when the number of its connections drops below its lower connection threshold. If Minimum Connections is not set but Maximum Connections is set, the server will receive new connections when the number of its connections drops below three fourths of its upper connection threshold.
9. 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.

DR Mode Considerations

The ARP Problem

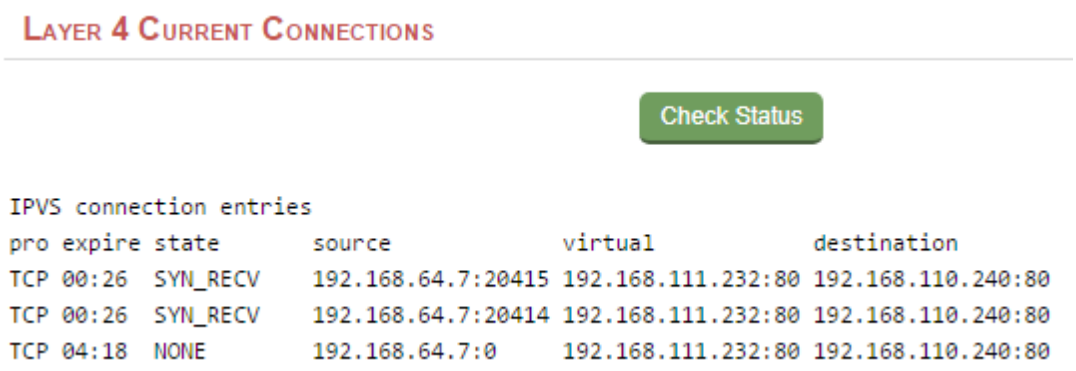
DR mode works by changing the MAC address of the inbound packets to match the Real Server selected by the load balancing algorithm. To enable DR mode to operate:

1. Each Real Server must be configured to accept packets destined for both the VIP address **and** the Real Server's 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). The service/process (e.g. IIS) must also respond to both addresses.
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 referred to as ***'Solving the ARP Problem'***. The steps required depend on the OS used as detailed in the following sections.

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.



If it is, this is normally a good indication that the ***ARP Problem*** has not been correctly solved.

Solving the ARP Problem for Linux

Method 1 (using iptables)

You can use iptables (netfilter) on each Real Server to re-direct incoming packets destined for the Virtual Service IP address. To make this permanent, simply add the following command to an appropriate start-up script such as /etc/rc.local on each of your Real Servers. If Real Servers are serving multiple VIPs, add additional iptables rules for each VIP.

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

e.g.

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

Note	Change the IP address to be the same as your Virtual Service.

This means 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.

Note	Method 1 may not always be appropriate if you're using IP-based virtual hosting on your web server. This is because the iptables rule above redirects 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 2 below instead.

Note	Method 1 does not work with IPv6 Virtual Services, use method 2 below instead.

Method 2 (using arp_ignore sysctl values)

This is the preferred method as it supports both IPv4 and IPv6. Each Real Server needs the loopback adapter to be configured with the Virtual Services IP address. This address must not respond to ARP requests and the web server also needs to be configured to respond to this address. To set this up follow steps 1-4 below on each Real Server.

Step 1 of 4: re-configure ARP on the Real Servers (this step can be skipped for IPv6 Virtual Services)

To do this add the following lines to /etc/sysctl.conf:

```
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
```

Note | Adjust the commands shown above to suit the network configuration of your servers.

Step 2 of 4: re-configure DAD on the Real Servers (this step can be skipped for IPv4 Virtual Services)

To do this add the following lines to /etc/sysctl.conf:

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

Step 3 of 4: apply these settings

Either reboot the Real Server or run the following command to apply these settings:

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

Step 4 of 4: add the Virtual Services IP address to the loopback adapter

Run the following command for each VIP. To make this permanent, simply add the command to an appropriate startup script such as /etc/rc.local.

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

for IPv6 addresses use:

```
ip addr add dev lo <Ipv6-VIP>/128
```

Note | You can check if this command added the VIP successfully using the command:

```
ip addr ls
```

You can remove the VIP from the loopback adapter using the command:

```
ip addr del dev lo <Ipv4-VIP>/32
```

Note

Steps 1, 2 & 3 can be replaced by writing directly to the required files using the following commands (run as root at the command line), this is temporary until the next reboot :

```
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
```

Method 3 (using firewalld)

In some newer versions of Linux, iptables is being deprecated in favour of firewalld. The following command can be used on each Real Server to resolve the ARP issue using firewalld:

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

e.g.

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

Note

Change the IP address to be the same as your Virtual Service.

To apply the new configuration, reload the firewall rules:

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

The current permanent configuration will become the new firewalld runtime configuration as well as the configuration at the next system start.

Solving the ARP Problem for Solaris

With Solaris the loopback interface does not respond to ARP requests so you just add your VIPs to it:

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

1. 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 is 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 mistake in DR mode configurations.

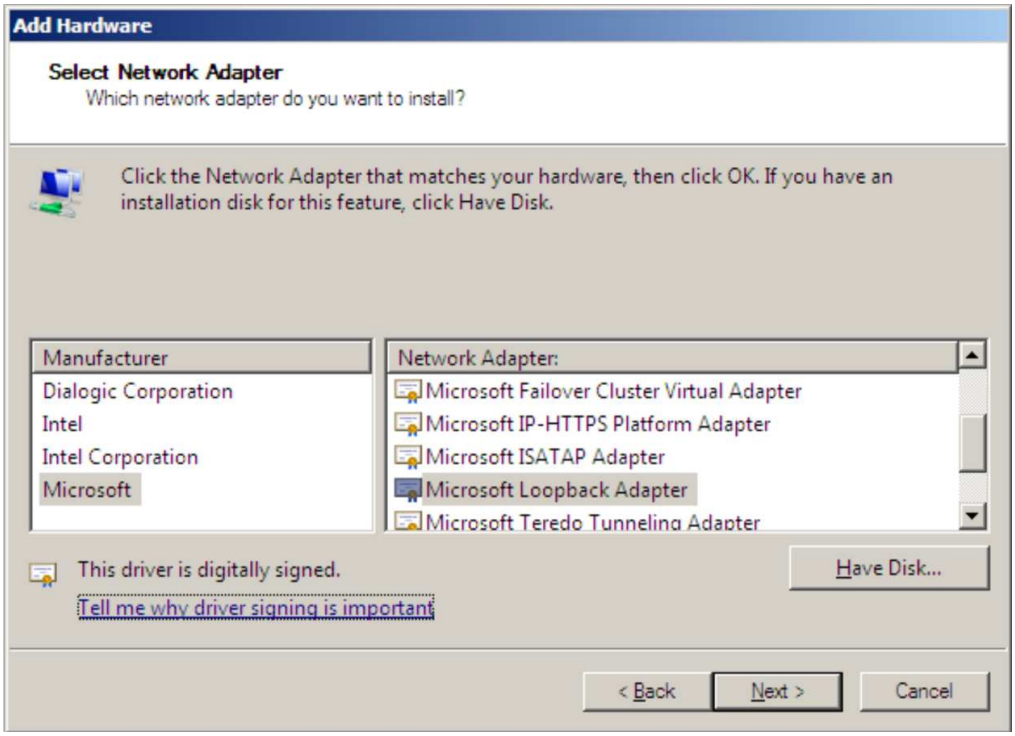
Solving the ARP Problem for Windows Servers

Windows Server 2008

Windows Server 2008 supports Direct Routing (DR) mode through the use of the Microsoft Loopback Adapter. The IP address allocated to the Loopback Adapter must be the same as the Virtual Service (VIP) address. If the Real Server is included in multiple DR mode VIPs, additional IP addresses can be added to the Loopback Adapter that correspond to each VIP. In addition, steps must be taken to set the strong/weak host behavior which is used to either block or allow interfaces to receive packets destined for a different interface on the same server.

Step 1 of 3: Install the Microsoft Loopback Adapter

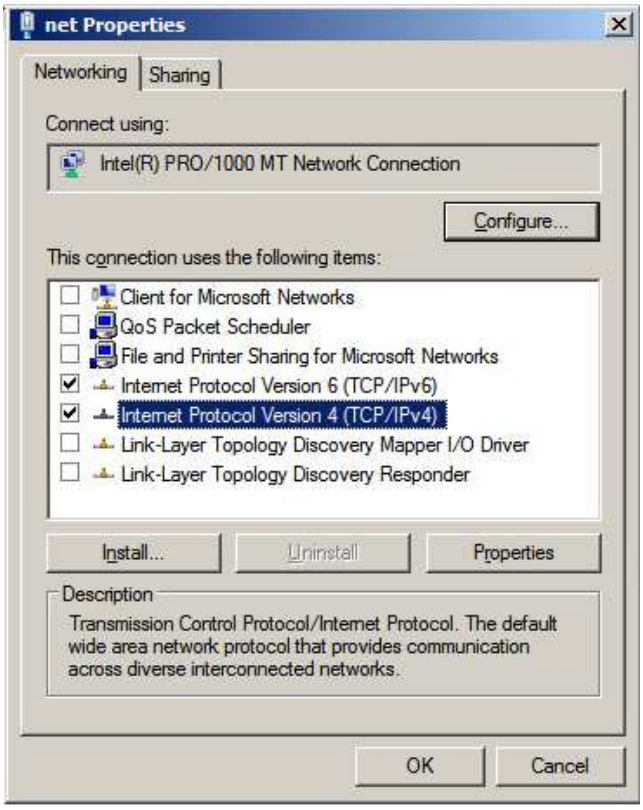
1. Click **Start**, then run **hdwwiz** to start the Hardware Installation Wizard.
2. When the Wizard has started, click **Next**.
3. Select **Install the hardware that I manually select from a list (Advanced)**, click **Next**.
4. Select **Network adapters**, click **Next**.
5. Select **Microsoft**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



6. Click **Next** to start the installation, when complete click **Finish**.

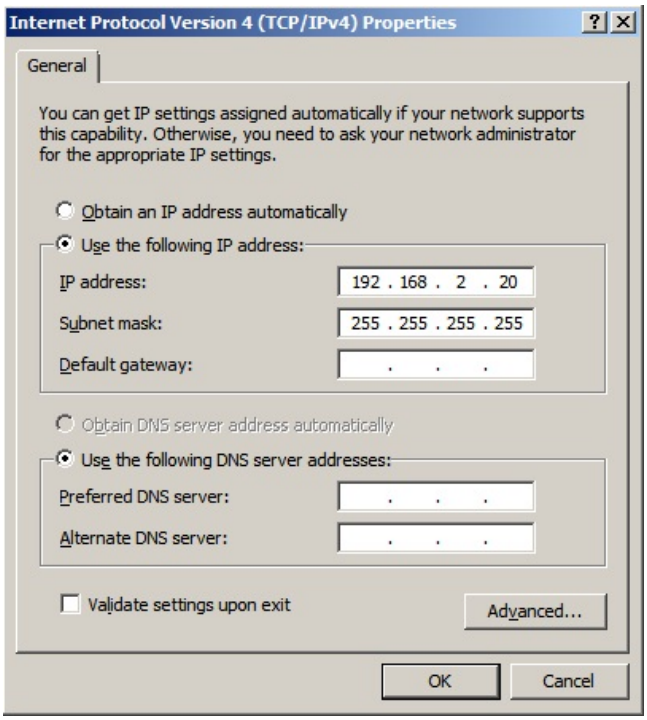
Step 2 of 3: Configure the Loopback Adapter

1. Open Control Panel and click **View Network status and tasks** under **Network and internet**.
2. Click **Change adapter settings**.
3. Right-click the new Loopback Adapter and select **Properties**.
4. Uncheck all items except **Internet Protocol Version 4 (TCP/IPv4)** and **Internet Protocol Version 6 (TCP/IPv6)** as shown below:



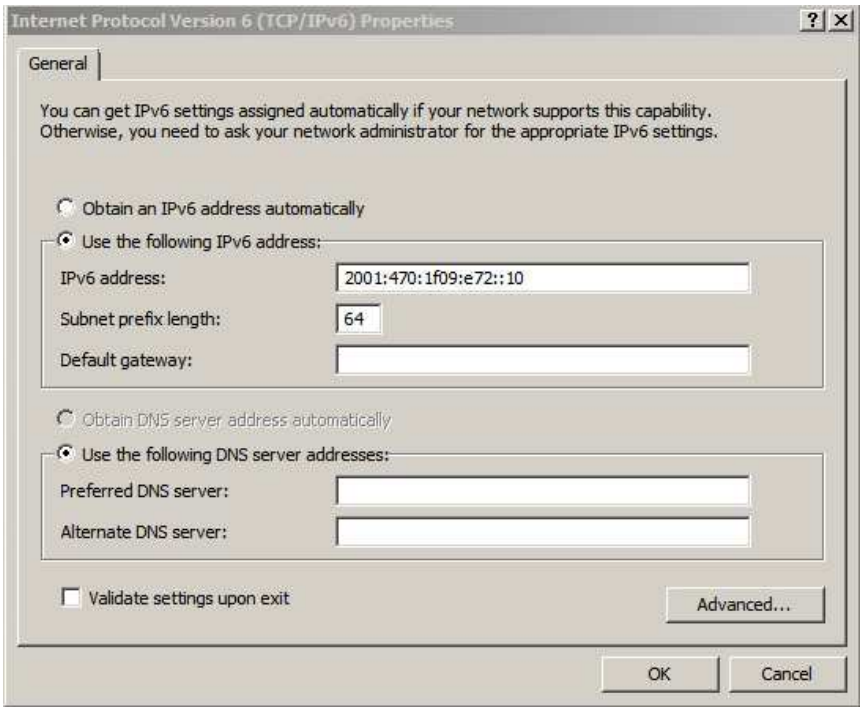
Note	Leaving both checked ensures that both IPv4 and IPv6 are supported. Select one If preferred.

5. If configuring IPv4 addresses select **Internet Protocol Version (TCP/IPv4)**, click **Properties** and configure the IP address to be the same as the Virtual Service (VIP) with a subnet mask of 255.255.255.255 , e.g. 192.168.2.20/255.255.255.255 as shown below:



6. If configuring IPv6 addresses select **Internet Protocol Version (TCP/IPv6)**, 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:



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

Note	For Windows 2008, it's not necessary to modify the interface metric on the advanced tab and should be left set to Automatic.

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

To configure the correct strong/weak host behavior for Windows 2008, the following commands must be run on each Real Server:

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 these commands to work, the LAN connection NIC must be named "net" and the loopback NIC must be named "loopback" as shown below. If you prefer to leave your current NIC names, then the commands above must be modified accordingly. For example, if your network adapters are named "LAN" and "LOOPBACK", the commands required would be:

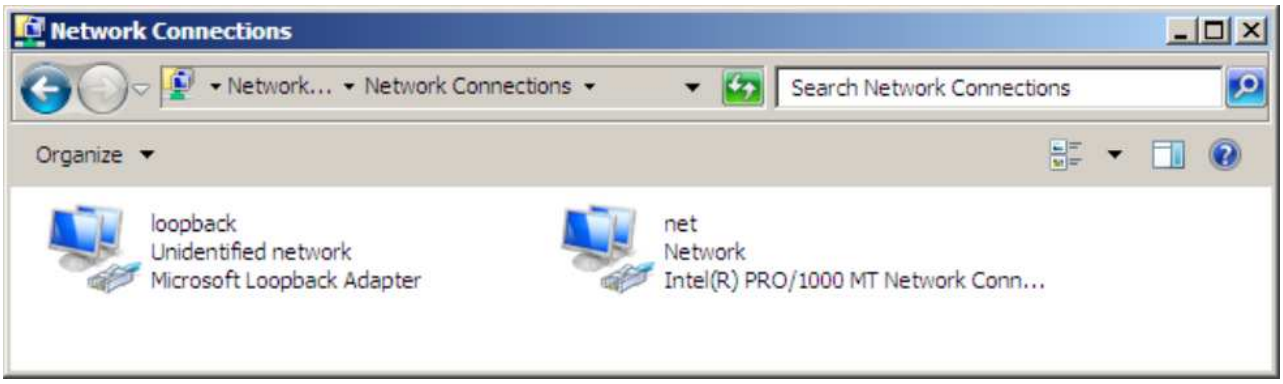
```
netsh interface ipv4 set interface "LAN" weakhostreceive=enabled
netsh interface ipv4 set interface "LOOPBACK" weakhostreceive=enabled
netsh interface ipv4 set interface "LOOPBACK" weakhostsend=enabled
```

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
```

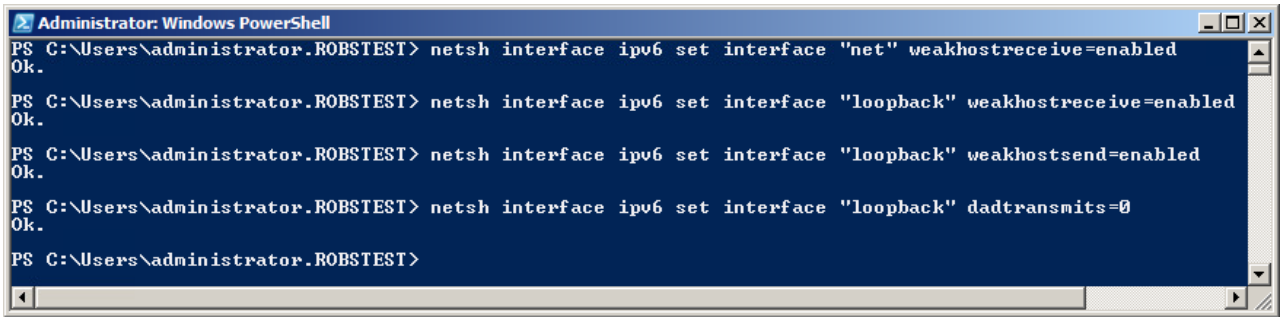
For these commands to work, the LAN connection NIC must be named "net" and the loopback NIC must be named "loopback" as shown below. If you prefer to leave your current NIC names, then the commands above must be modified accordingly. For example, if your network adapters are named "LAN" and "LOOPBACK", the commands required would be:

```
netsh interface ipv6 set interface "LAN" 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
```



Note | The names for the NICs are case sensitive, so make sure that the name used for the interface and the name used in the commands match exactly.

- Start PowerShell or use a command window to run the appropriate netsh commands as shown in the example below:



Note | This shows an IPv6 example, use the IPv4 commands if you're using IPv4 addresses.

Repeat steps 1 - 3 on all remaining Windows 2008 Real Server(s).

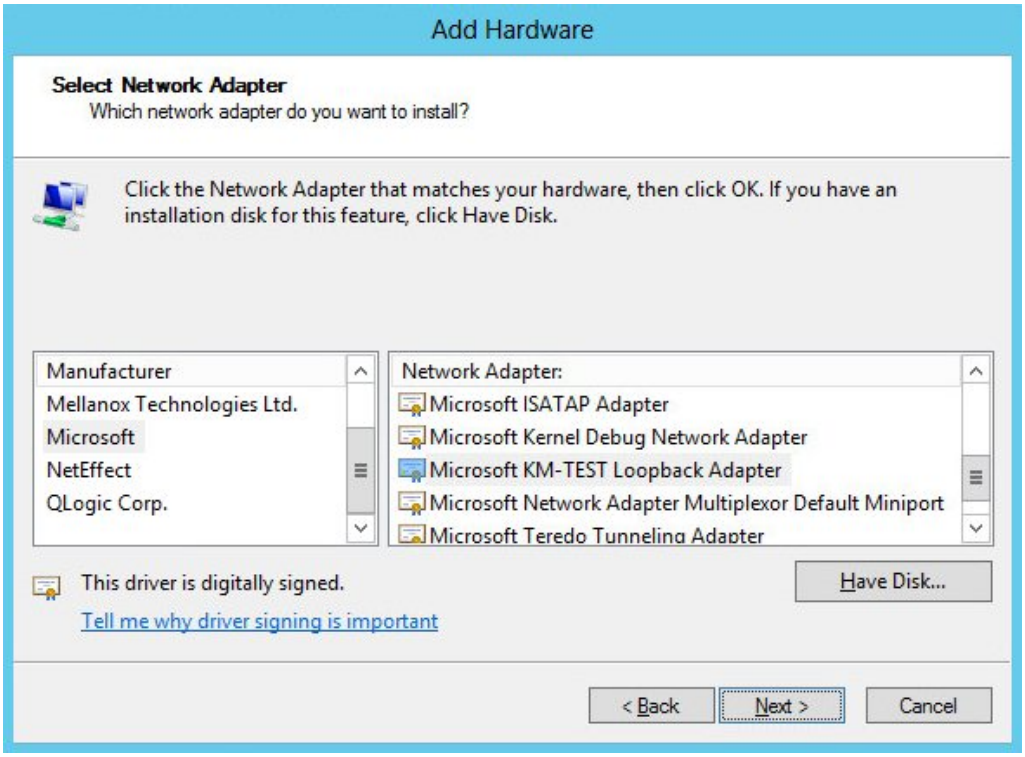
Windows Server 2012 & Later

Windows Server 2012 and later support Direct Routing (DR) mode through the use of the Microsoft Loopback Adapter. The IP address allocated to the Loopback Adapter must be the same as the Virtual Service (VIP) address. If the Real Server is included in multiple DR mode VIPs, additional IP addresses can be added to the Loopback Adapter that correspond to each VIP. In addition, steps must be taken to set the strong/weak host behavior which

is used to either block or allow interfaces to receive packets destined for a different interface on the same server.

Step 1 of 3: Install the Microsoft Loopback Adapter

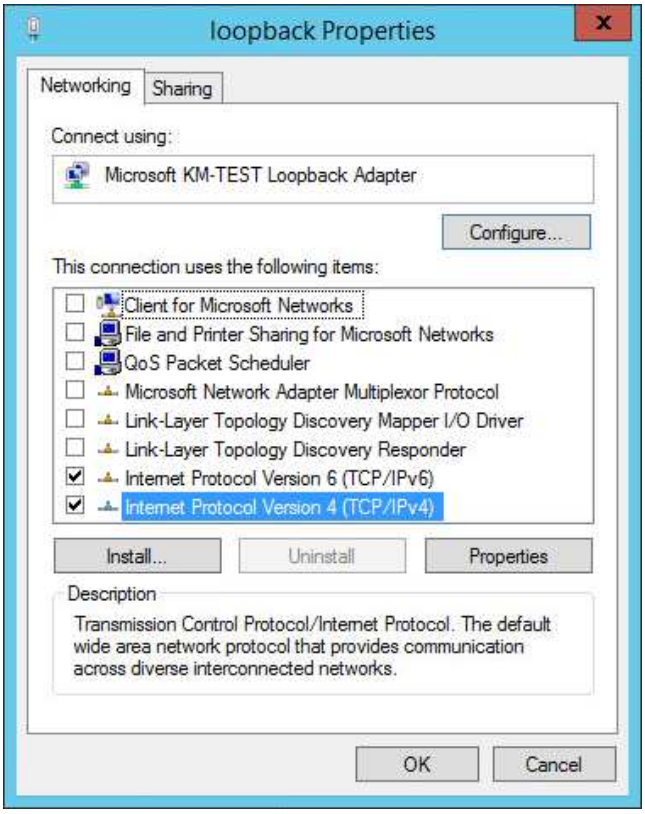
1. Click **Start**, then run **hdwwiz** to start the Hardware Installation Wizard.
2. When the Wizard has started, click **Next**.
3. Select **Install the hardware that I manually select from a list (Advanced)**, click **Next**.
4. Select **Network adapters**, click **Next**.
5. Select **Microsoft** & **Microsoft KM-Test Loopback Adapter**, click **Next**.



6. Click **Next** to start the installation, when complete click **Finish**.

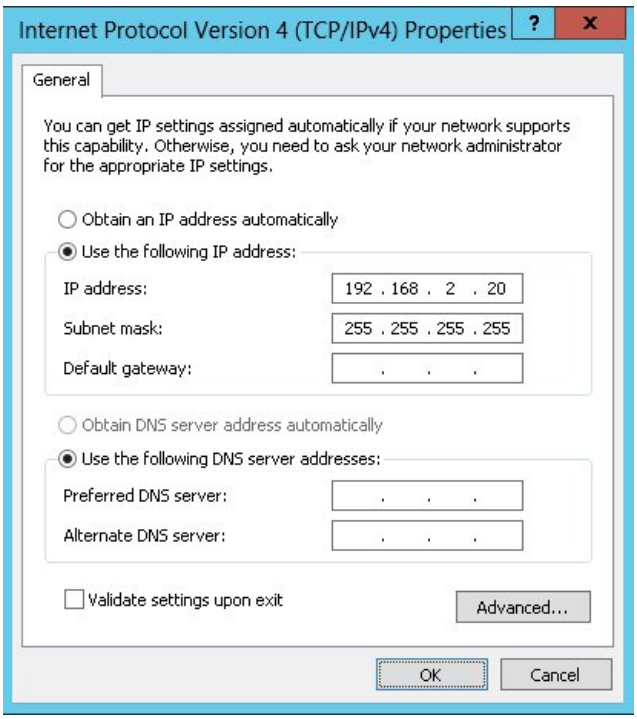
Step 2 of 3: Configure the Loopback Adapter

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**.
4. Uncheck all items except **Internet Protocol Version 4 (TCP/IPv4)** and **Internet Protocol Version 6 (TCP/IPv6)** as shown below:



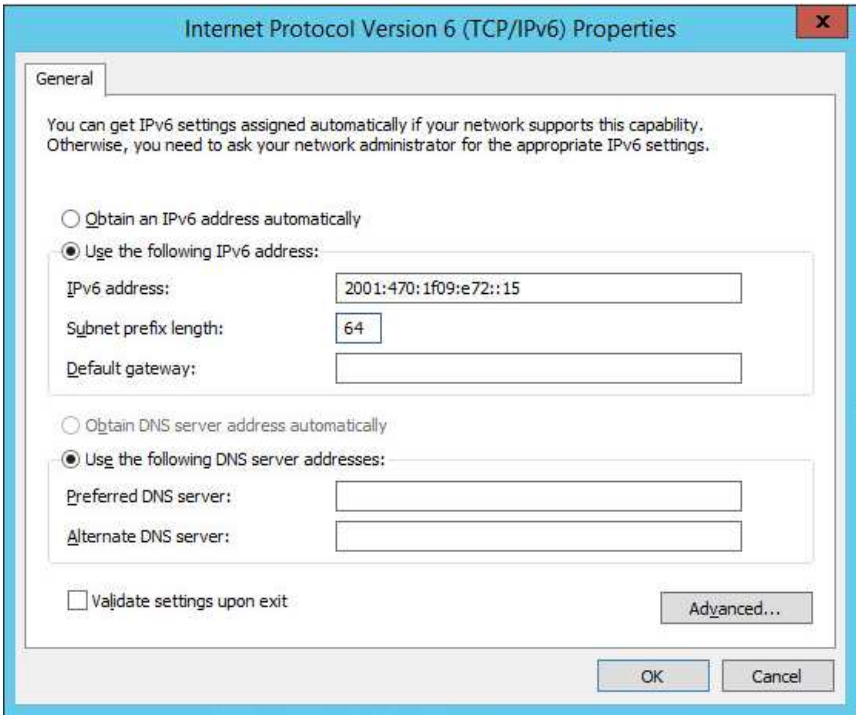
Note	Leaving both checked ensures that both IPv4 and IPv6 are supported. Select one If preferred.

5. If configuring IPv4 addresses select **Internet Protocol Version (TCP/IPv4)**, click **Properties** and configure the IP address to be the same as the Virtual Service (VIP) with a subnet mask of 255.255.255.255 , e.g. 192.168.2.20/255.255.255.255 as shown below:



6. If configuring IPv6 addresses select **Internet Protocol Version (TCP/IPv6)**, 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:



7. Click **OK** on TCP/IP Properties, then click **Close** on Ethernet Properties to save and apply the new settings.

Note

For Windows 2012/2016/2019, it's not necessary to modify the interface metric on the advanced tab and should be left set to Automatic.

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

To configure the correct strong/weak host behavior for Windows 2012/2016/2019, the following commands must be run on each Real Server:

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 these commands to work, the LAN connection NIC must be named "net" and the loopback NIC must be named "loopback" as shown below. If you prefer to leave your current NIC names, then the commands above must be modified accordingly. For example, if your network adapters are named "LAN" and "LOOPBACK", the commands required would be:

```
netsh interface ipv4 set interface "LAN" 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
```

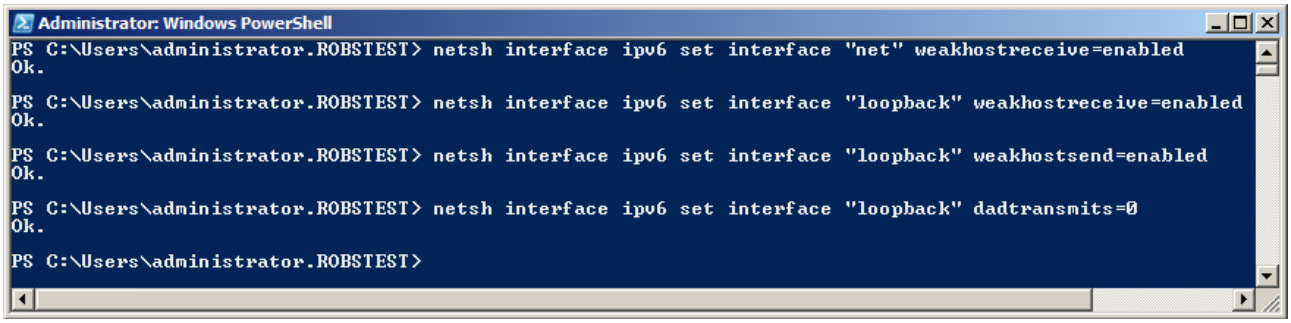
For these commands to work, the LAN connection NIC must be named "net" and the loopback NIC must be named "loopback" as shown below. If you prefer to leave your current NIC names, then the commands above must be modified accordingly. For example, if your network adapters are named "LAN" and "LOOPBACK", the commands required would be:

```
netsh interface ipv6 set interface "LAN" 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
```

Note

The names for the NICs are case sensitive, so make sure that the name used for the interface and the name used in the commands match exactly.

- Start PowerShell or use a command window to run the appropriate netsh commands as shown in the example below:



Note

This shows an IPv6 example, use the IPv4 commands if you're using IPv4 addresses.

Repeat steps 1 - 3 on all remaining Windows 2012/2016/2019 Real Server(s).

If preferred you can also use the following PowerShell Cmdlets:

The following example configures both IPv4 and IPv6 at the same time:

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

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

To configure just IPv4:

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

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

To configure just IPv6:

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

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

Verifying Strong/Weak Host Settings

To verify that settings have been configured correctly, run the following command on each Real Server to clearly list the settings that have been applied to the interface:

```
netsh interface ipv4 show interface <interface name>
```

For the 'loopback' adapter run:

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

For the 'net' adapter run:

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

The following image shows the output for the loopback adapter:

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

Interface loopback Parameters
----------------------------------------------
IfLuid                             : ethernet_9
IfIndex                            : 15
State                              : connected
Metric                             : 30
Link MTU                           : 1500 bytes
Reachable Time                     : 28500 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

C:\Users\Administrator>
```

Note | For IPv6, simply 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 | For Windows 2008 R2 & later, if you want to leave the built-in firewall enabled, you'll either need to enable the relevant default firewall exceptions or create your own to enable access to the web server. These exceptions will allow traffic on both the network and loopback adapters.

Note | Failure to correctly configure the Real Servers to handle the ***ARP Problem*** is the most common problem in DR configurations.

Solving the ARP Problem - Possible Side Effect for Windows 2008 R2 & 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 following two PowerShell commands should be run on each Windows 2008 R2 & later Real Server to set the **SkipAsSource** flag for all IPs assigned to the loopback adapter:

```
[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

To verify that the flag has been set for all IPs, the following PowerShell command can be used:

```
Get-NetIPAddress -InterfaceAlias loopback
```

Note | For more information about these commands, please refer to this Microsoft article.

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. This can be done in 2 ways:

1) using the NIC's advanced properties tab, disable the following settings:

Recv Segment Coalescing (IPv4)
Recv Segment Coalescing (IPv6)

2) using PowerShell:

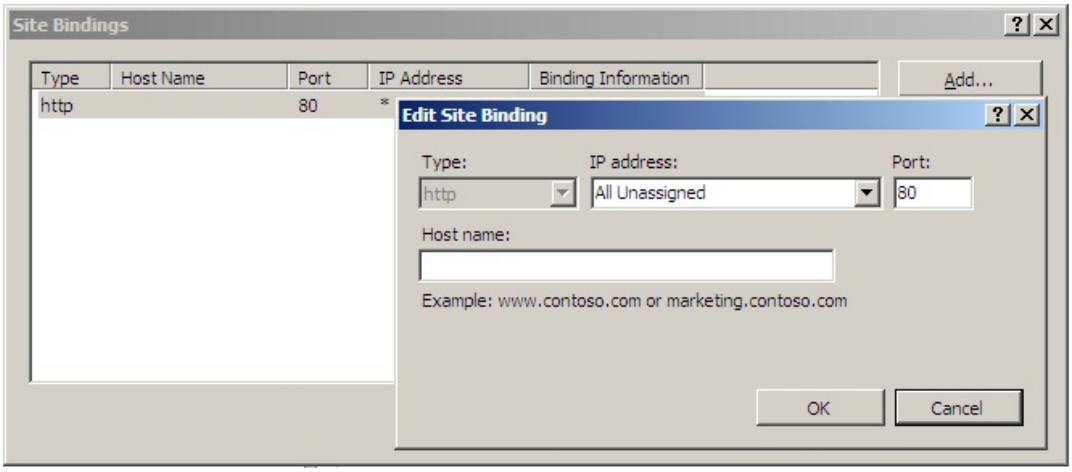
```
Disable-NetAdapterRsc -Name "MyAdapter" -IPv4
Disable-NetAdapterRsc -Name "MyAdapter" -IPv6
```

Configuring Your Application to Respond to Both the RIP and VIP

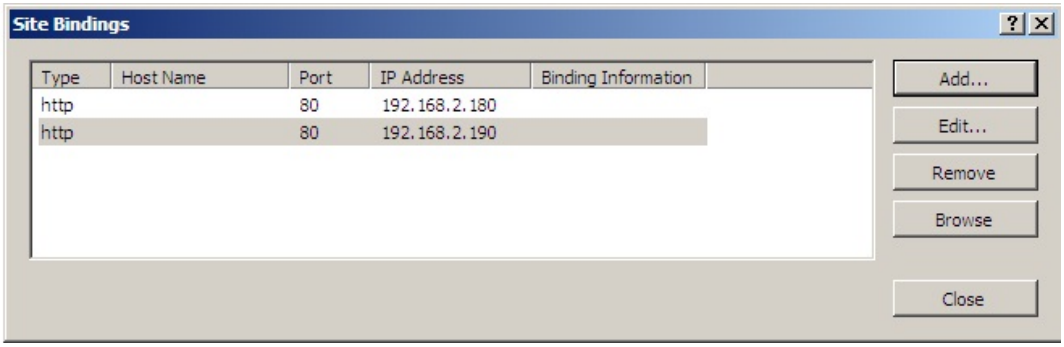
For DR & TUN modes, it's also important to make sure that your application (IIS in this example) responds to both the VIP and RIP.

Windows 2008 & Later

By default, IIS listens on all configured IP addresses, this is shown in the example below (shows Windows 2008 R2 example). As can be seen the IP address field is set to "All Unassigned".



If the default configuration is left, 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	These examples illustrates how IIS must be configured to ensure that its listening on both the RIP and VIP address. Remember that this applies equally to **all** applications when using DR mode.

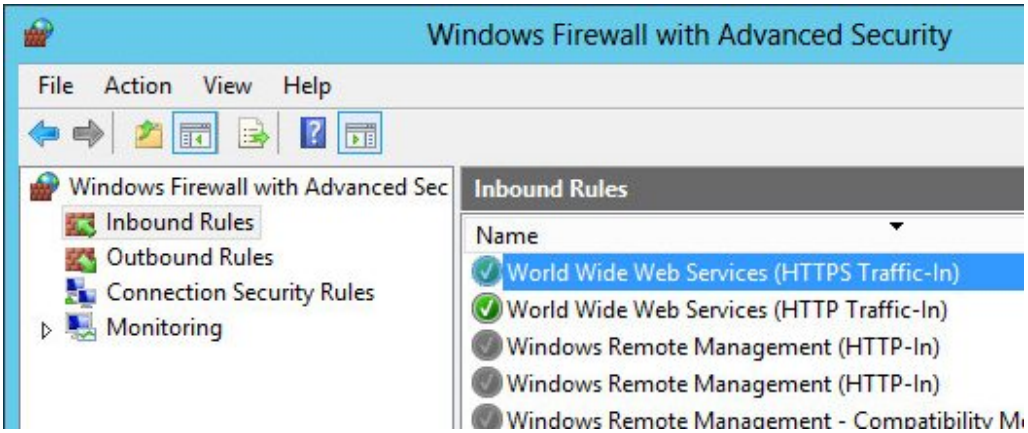
Windows Firewall Settings

Windows 2008 R2 & Later

Windows 2008 R2 and later automatically creates several default firewall rules for both inbound and outbound traffic. There are 3 firewall profiles and interfaces can be associated with one of these 3 profiles (domain, private and public) although the Loopback Adapter automatically gets associated with the public profile and this cannot be changed. For a web server listening on port 80 the following default HTTP rules need to be enabled as shown below:

Windows 2012 & Later

Windows 2012 is similar to Windows 2008 R2 as shown below:



NAT Mode Considerations

Layer 4 NAT mode requires Real Server return traffic to pass back via the load balancer. This is achieved by setting the Real Server's default gateway to be the load balancer. For an HA Pair, an additional floating IP address should be used to allow failover. 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 specific services to be exposed on your Real Servers you have two choices:

- Setup a Virtual Service with a single Real Server for each service ***Or***
- Setup a floating IP address and individual SNAT/DNAT rules for each service as shown in the example below. These lines can be added to the firewall script using the WebUI menu option *Maintenance > Firewall Script*.

```
INT_ADDR="192.168.110.240"
EXT_ADDR="10.200.110.240"
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 the above SNAT/DNAT rules have been configured, the following firewall entries will be listed under *View Configuration > Firewall Rules:*

```
Chain PREROUTING (policy ACCEPT 2 packets, 120 bytes)
pkts bytes target prot opt in out source destination
0 0 DNAT tcp -- * * 0.0.0.0/0 10.200.110.240 to:192.168.110.240
Chain POSTROUTING (policy ACCEPT 1 packets, 60 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
```

Note | The default gateway on the Real Server must be an IP on the load balancer.

Note | If Autonat is already enabled, only the DNAT rule (i.e. not the SNAT rule) will be required.

Note | Please don't hesitate to contact support@loadbalancer.org to discuss any specific requirements you may have.

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

Ensure you specify your local subnet address. Replace "192.168.2.21" with the IP address of your load balancer. Replace "LAN" with the name of your Interface.

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. Replace 192.168.2.21 with the IP address of your load balancer.

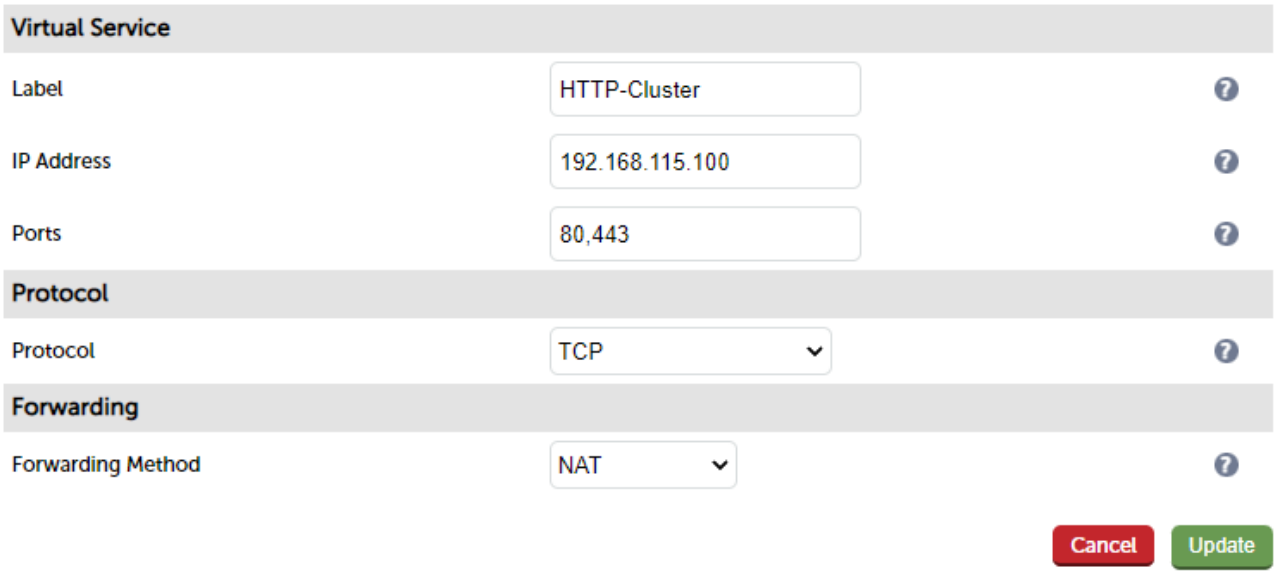
Any local traffic (same subnet) is then handled by the manual route and any external traffic is handled by the default route (which also points at the load balancer).

Firewall Marks

Using firewall marks enables multiple ports and/or multiple IP addresses to be combined into a single Virtual Service. A common use of this feature is to aggregate port 80 (HTTP) and port 443 (HTTPS) so that when a client fills their shopping cart via HTTP, then moves to HTTPS to give their credit card information, they will remain on the same Real Server.

Firewall Marks - Auto Configuration

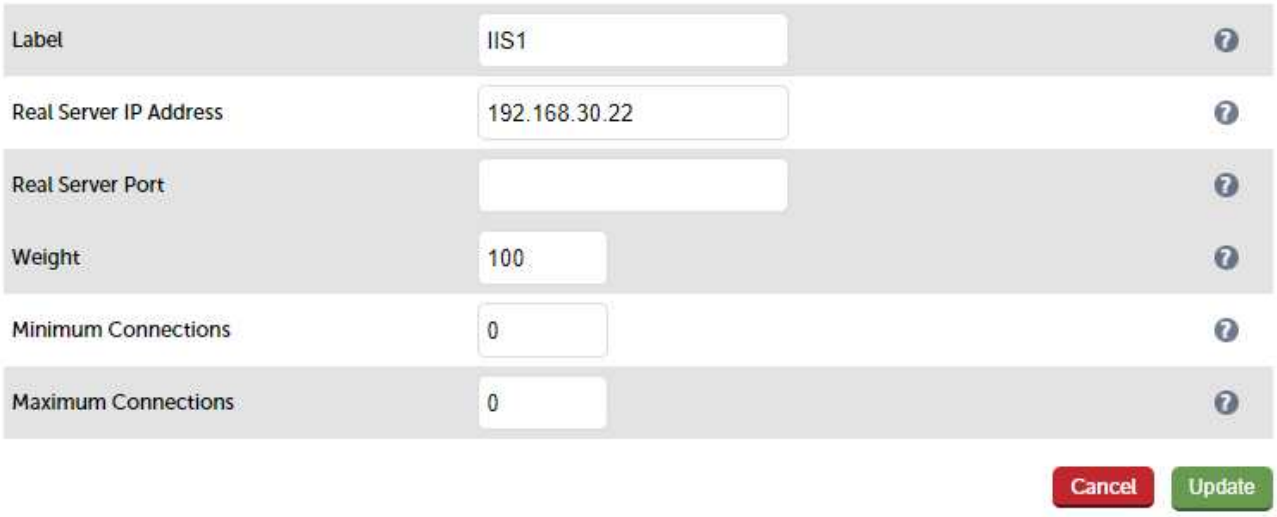
When defining a layer 4 VIP with multiple ports, firewall marks are used automatically in the background to enable this functionality. For example, to configure an HTTP & HTTPS NAT mode Virtual Service, port 80 & 443 must be specified separated by a comma in the 'Virtual Service Ports' field as shown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



This will automatically configure the load balancer for firewall marks.

Note	Persistence will be enabled automatically.

For NAT mode VIPs, leave the *Real Server Port* field blank as shown below:



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. Packets will be forwarded to the same port as specified for the VIP.

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

Note	The Health check port is automatically set to be the first port in the list, e.g. if ports 80 & 443 are defined for the VIP, the check port is automatically set to port 80. This can be changed if required using the *Check Port* field.

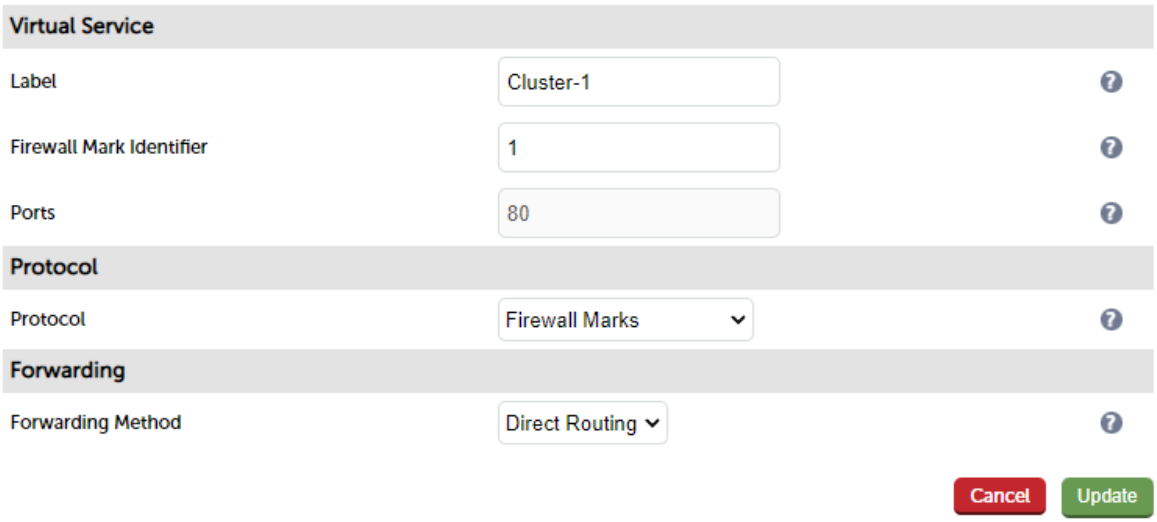
Firewall Marks - Manual Configuration

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 and mark them with an arbitrary integer. A Virtual Service is also configured specifying this firewall mark integer instead of the IP address.

EXAMPLE 1 - Setup a new DR Mode Firewall Mark when no Initial VIP has been Created

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**.



3. Define 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 defining the firewall rules.
5. The *Virtual Service Ports* field does not need to be set as it is not relevant in this case - the actual port(s) used are defined in the firewall script in step 5 below.
6. Set *Protocol* to **Firewall Marks** - at this point the *Virtual Service Ports* field will be grayed out and the Virtual Service *IP Address* field will be renamed as *Firewall Mark Identifier* as shown above.
7. Click **Update**.

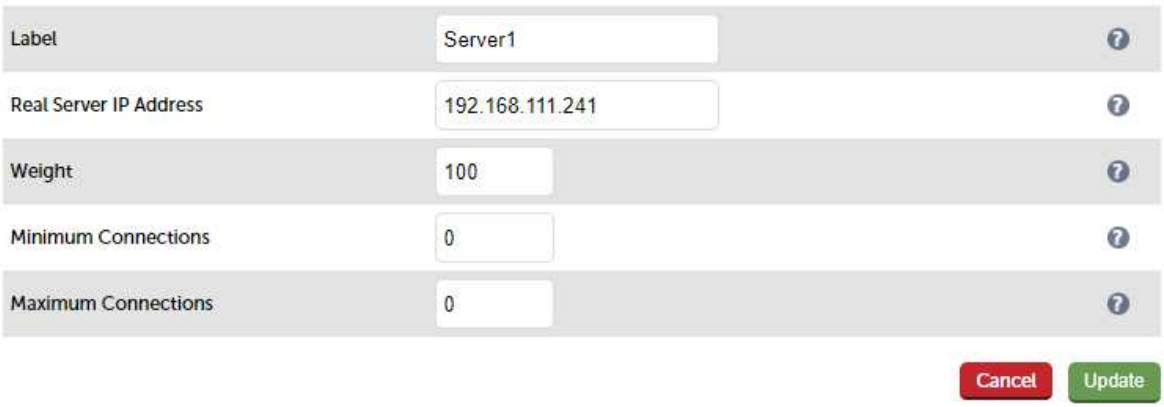
Note	Persistence will be enabled automatically.

Step 2: Define 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

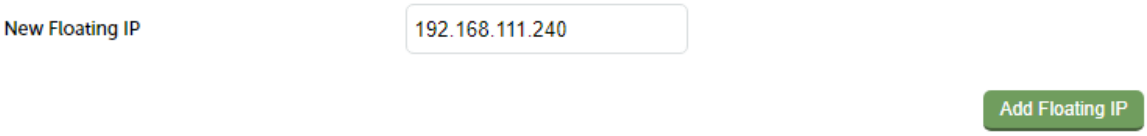
1. Using the WebUI, navigate to: *Cluster Configuration > Layer 4 - Real Servers*.
2. Click **Add a new Real Server**.
3. Enter the required details as shown below.



4. Click **Update**.

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

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**.



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

```
27
28  ########################  Manual Firewall Marks  ############################
29
30  # Example: Associate HTTP and HTTPS with Firewall Mark 1:
31  #VIP1="10.0.0.66"
32  #iptables -t mangle -A PREROUTING -p tcp -d $VIP1 --dport 80 -j MARK --set-mark 1
33  #iptables -t mangle -A PREROUTING -p tcp -d $VIP1 --dport 443 -j MARK --set-mark 1
34
35  # A Virtual Service may then be created in the web interface, using 1 as the
36  # service address.
37
38  #It is also possible to bind TCP and UDP protocols together with a firewall mark.
39  #VIP1="192.168.64.27"
40  #iptables -t mangle -A PREROUTING -p tcp -d $VIP1 --dport 80 -j MARK --set-mark 1
41  #iptables -t mangle -A PREROUTING -p udp -d $VIP1 --dport 300 -j MARK --set-mark 1
42
43  VIP1="192.168.111.240"
44  iptables -t mangle -A PREROUTING -p tcp -d $VIP1 --dport 8025 -j MARK --set-mark 1
45  iptables -t mangle -A PREROUTING -p udp -d $VIP1 --dport 8025 -j MARK --set-mark 1
46
47  ##########################  Packet Filtering  ################################
48
49  # You should always use a network perimeter firewall to lock down all
50  # external access to the load balancer except the required Virtual Services
51  # and the required services from your admin machine / network (SSH & HTTPS)
52
53  # Allow unlimited traffic on the loopback interface:
54  #iptables -A INPUT -i lo -j ACCEPT
55  #iptables -A OUTPUT -o lo -j ACCEPT
56
57
58  #Do not delete the following 2 lines.
    echo "Firewall Activated"
```

Update

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

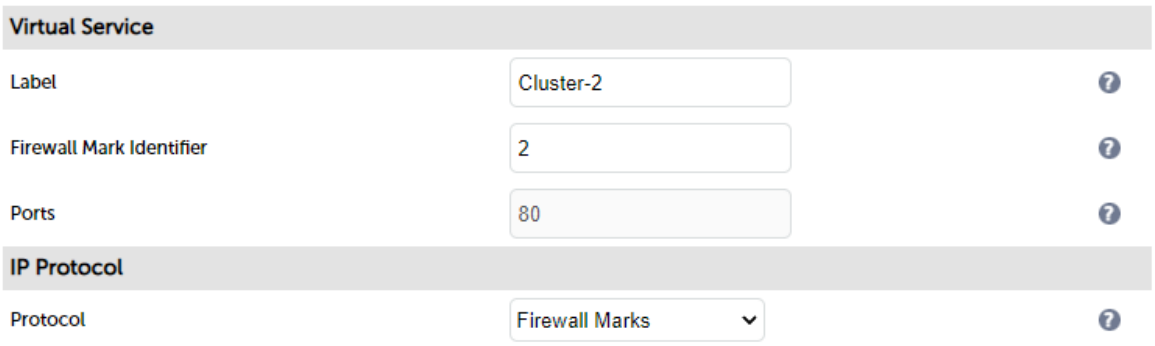
The VIP is now configured and will be accessible on 192.168.111.240, TCP & UDP port 8025

EXAMPLE 2 - Setup a Firewall Mark by Modifying an Existing VIP

In this case, the floating IP address associated with the VIP will already exist so does not need to be created manually.

Step 1: Modify the Existing Virtual Service

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



3. Change the IP address to the chosen 'mark' value as shown above, e.g. **2**.

4. change the *Protocol* field to **Firewall Marks**.

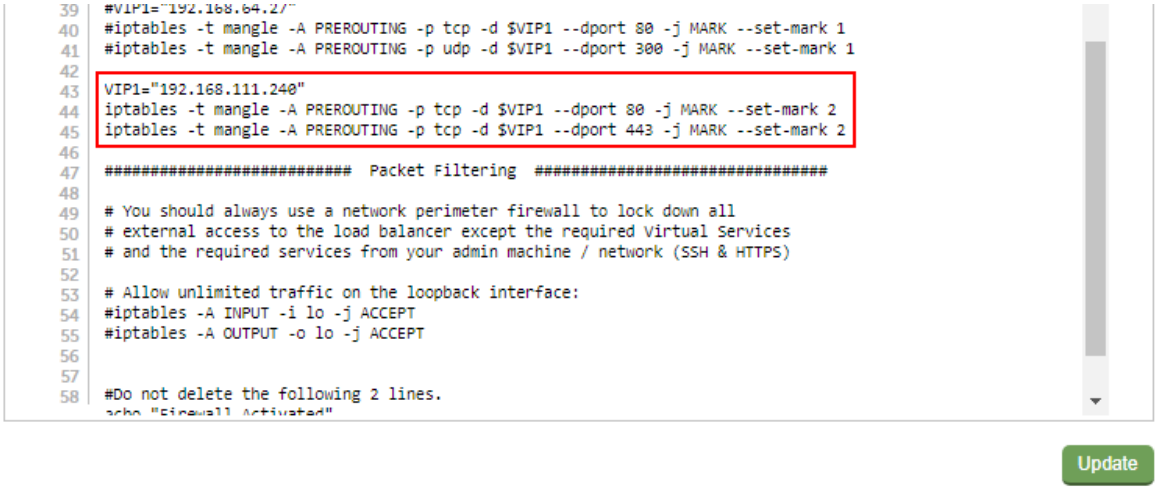
Step 2: Define a health check Port

1. Enter the appropriate value in the *Check Port* field, e.g. **80**.
2. Click **Update**.

Step 3: Modify the Firewall Script

1. Using the WebUI, navigate to: *Maintenance > Firewall Script*.
2. Enter the rules to configure the Firewall Mark as shown in the example below:

```
VIP1="192.168.111.240"
iptables -t mangle -A PREROUTING -p tcp -d $VIP1 --dport 80 -j MARK --set-mark 2
iptables -t mangle -A PREROUTING -p tcp -d $VIP1 --dport 443 -j MARK --set-mark 2
```



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

The VIP is now configured and will be accessible on 192.168.111.240, TCP ports 80 & 443

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. You can only have one health check port assigned, so if you are grouping port 80 and 443 traffic together you can only check one of these ports, typically this would be port 80.
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)

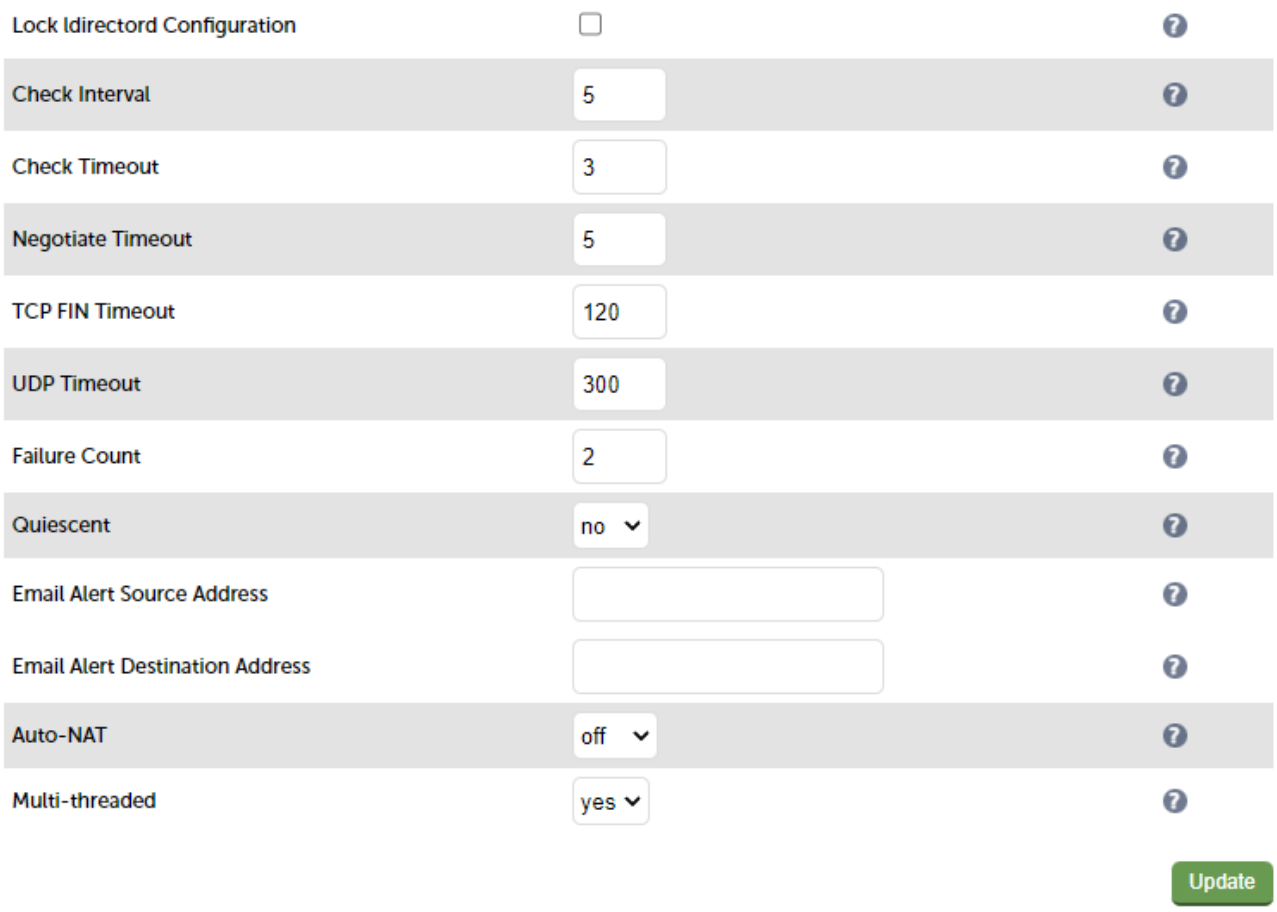
5. You can specify a range of IP addresses as shown below:

```
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
```

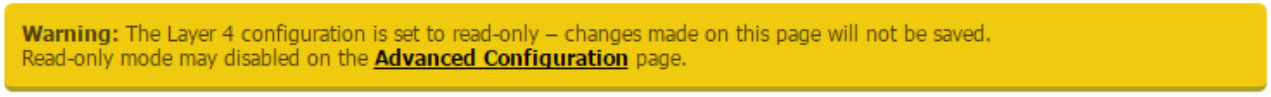
(this specifies the destination IP address as a range from 10.141.12.34 to 10.141.12.40)

Layer 4 - Advanced Configuration

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



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.



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

Check Timeout - Layer 4 (Ldirectord) 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 (Ldirectord) 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 (Ldirectord) 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 as 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 and HTTP-based 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.

Layer 7 supports a number of persistence methods including source IP address, HTTP cookie (both application based and inserted), Connection Broker, RDP cookie and SSL session ID.

When a VIP is added the load balancer automatically adds a corresponding floating IP address which is activated instantly. Check *View Configuration > Network Configuration* to ensure that the Floating IP address has been activated correctly. They will show up as secondary IP addresses under the relevant interface.

Multiple ports can be defined per VIP, for example 80 & 443. In this persistence (aka affinity/stickiness) 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 possible 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 (VIPs)

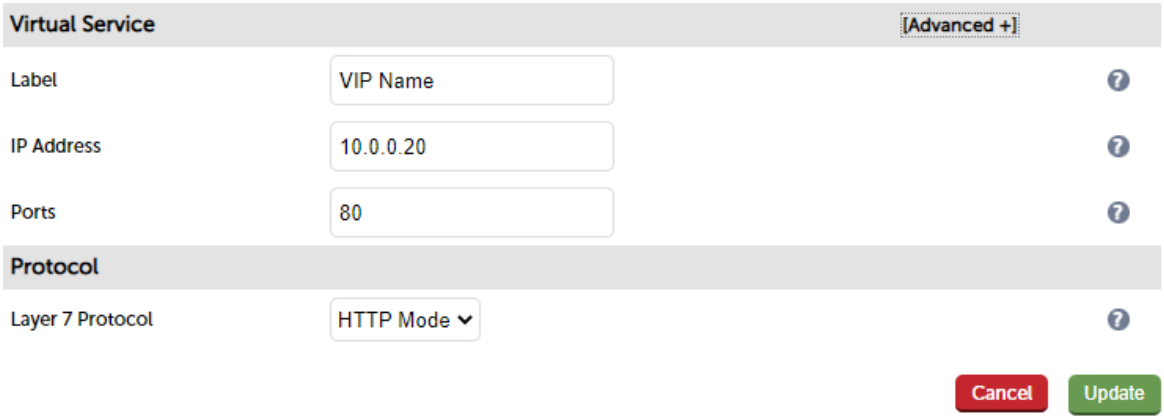
Virtual services can be created in 2 ways, either by defining a new VIP from scratch where the required settings must be defined manually, or by using the duplicate VIP feature.

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'd create a VIP for a web cluster, another for an FTP cluster and a third for a SIP cluster. Multiple ports can also be specified for each VIP.

Defining a New Layer 7 VIP

to add a new layer 7 VIP:

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Virtual Services*.
2. Click **Add a new Virtual Service**.



3. Enter an appropriate *Label* (name) for the new Virtual Service.
4. Enter the required IP address in the *Virtual Service IP Address* field.
5. Enter the required ports(s) in the *Virtual Service Ports* field, separate multiple ports with commas, specify a range with a hyphen.

> Note
>
> Several ports are used by the appliance and therefore cannot be used for Virtual Services. For full details please refer to Ports Used by the Appliance.

6. Select the Layer 7 protocol to be handled by this Virtual Service, either HTTP or TCP.
   - **HTTP Mode** - Selected if the Virtual Service will handle only HTTP traffic. Allows more flexibility in the processing of connections. The HTTP Cookie and HTTP application cookie modes, and the X-Forwarded-For header all require HTTP to be selected. In addition, HAProxy logs will show more information on the client requests and Real Server responses.
   - **TCP Mode** - Required for non HTTP traffic such as HTTPS, RPC, RDP, FTP etc.
7. Click **Update**.
8. Now proceed to define the RIPs (Real Servers) as detailed here.

Duplicating an Existing Layer 7 VIP

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

> Note
>
> This option will copy all Virtual Service settings along with all associated Real Servers. After duplicating, you'll need to change either the IP address or port. If this is not done, the new VIP will clash with the original VIP and will not load. All other settings can remain the same if required.

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 first adding a Virtual Service, only certain values can be configured, others are set at their default value to simplify initial configuration. These values can be changed after the Virtual Service has been created by clicking **Modify** next to the relevant Virtual Service. Additional settings that can be changed are:

Section	Setting	Description
Virtual Service	***Advanced*** > Manual Configuration	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 labels match the one 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. Note: For more information on creating a manually defined layer 7 VIP, please refer to Layer 7 - Custom Configurations.
	Advanced > Create Backend Only	Enabling this option will stop 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. Note: For more information on creating backend VIPs please refer to HAProxy Backends.
Protocol	***Advanced*** > HTTP Pipeline Mode (HTTP mode only)	Select how HAProxy should handle HTTP pipelining to client and server. The options are: • **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 using HTTP. • **Keep-alive client, close server** - Allow client to negotiate pipelining, whilst closing the server connection using HTTP. • **Close client, force close server** - Close the server connection at the TCP layer, as well as sending the Connection: close header. Also close the client connection using HTTP.

Section	Setting	Description
	Advanced > Work around broken *Connection: close* (HTTP mode only)	Work around Real Servers that do not correctly implement the HTTP *Connection:close* option.
	Advanced > Accept Invalid HTTP Requests (HTTP mode only)	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.
	Advanced > HTTP request timeout (DoS Protection) (HTTP mode only)	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.
	Advanced > Reuse Idle HTTP Connections (HTTP mode only)	It is possible to reuse idle connections to serve requests from the same session which can be beneficial in terms of performance. It is 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 > TCP Keep-alive (TCP Mode only)	Enables the transmission of TCP keep-alive on both the client and the server sides of the connection. Its 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 > Redispatch	If a real server becomes un-responsive ignore persistence and send client connection to another available real server. If Unsure leave enabled.

Section	Setting	Description
Connection Distribution Method	Balance Mode	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: • **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. The default weight for new Real Servers is 100. • **Weighted Least Connection (this is the default for new VIPs)** - 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. The default weight for new Real Servers is 100. *This is the default method for new VIPs*.

Section	Setting	Description
Persistence	Persistence Mode	Select how the load balancer should track clients so as to direct each request to the same server. The options are: • **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 form the request. Note: You cannot use the set X-Forwarded-For header option with this method of persistence. It will be disabled. • **Stick On Fallback** (An external configured fallback server is required) – This option disables automatically failing back to the Real Server from the fallback server when the Real Server comes back online. 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. • **None** - No persistence. The allocation of clients to Real Servers will be determined solely by the Balance Mode.
Persistence Options	HTTP Cookie Name	Set the name of the HTTP cookie.

Section	Setting	Description
	Application Cookie Name	Set the name of the application cookie.
	Advanced > HTTP Cookie Max Idle Duration	Set the max idle time of the cookie.
	Advanced > HTTP Cookie Max Life Duration	Set the max lifetime of the cookie.
	Advanced > Persistence timeout	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 > Persistence table size	The size of the table of connections in KB. The size of the table of connections (approx 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.
	Advanced > Clear Stick on Drain	Clearing the stick table 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. This will force users onto another node when they attempt to reconnect and the while server they were attached to is in drain mode. Alternatively disabling this option would allow the user to reconnect and they would only be moved when their persistence entry expired.

Section	Setting	Description
	Advanced > XFF IP Position	With XFF headers its 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. 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. It is possible to do the same thing with Multiple XFF headers: X-Forwarded-For: 192.168.1.1 X-Forwarded-For: 192.168.1.2 X-Forwarded-For: 10.10.10.1 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.

<table>
<tr><th>Section</th><th>Setting</th><th>Description</th></tr>
<tr><td>Health Checks</td><td>Check Type</td><td>
<table>
<tr><td>Note</td><td>For full details of all layer 7 health check options, please refer to Health Checks for Layer 7 Services.</td></tr>
</table>
Specify the type of health check to be performed on the Real Servers. The options are:

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 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:

<pre>USE mysql; INSERT INTO user (Host,User) values ('','');
FLUSH PRIVILEGES;</pre>

No checks, Always on - No health checks, all real servers are marked online.

</td></tr>
<tr><td>Health Check Options</td><td>Request to send</td><td>Specify a specific location/file for the health check. Open the specified location and check for the Response Expected. Useful for checking a server sided script to check the health of the backend application.</td></tr>
<tr><td></td><td>Response expected</td><td>The content expected for a valid health check on the specified file. The Response Expected can be any valid regular expression statement.</td></tr>
<tr><td></td><td>Advanced > Check Port</td><td>If specified, this setting overrides the default check port, useful when you are balancing multiple ports.</td></tr>
<tr><td></td><td>Advanced > Username</td><td>If authentication is required specify the username here.</td></tr>
</table>

Section	Setting	Description
	Advanced > Host Header	If the real server's web server is configured to require a Host header, the value to be used in health checks may be set here.
	Advanced > Password	If authentication is required specify the password here.
ACL Rules	Configure Content Redirects	Enables ACLs to be configured. For more information on configuring ACLs please refer to ACLs (aka Content Switching) and URL Rewriting.
Header Rules	Configure Headers	Enables HTTP headers to be added, set or deleted. For more information on configuring HTTP headers please refer to Modifying HTTP Header Fields.
Feedback Method	Feedback Method	Select whether HAProxy should query each Real Server for its load level. The options are: • **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	IP Address	IP address of server where to direct requests if all RIPs are down.
	Port	Port of server where to direct requests if all RIPs are down.
	Advanced > Fallback Persistence	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. Setting this to off will move users to a Real Server as soon as one is available.
	Advanced > Encrypt Connection	Enable SSL encryption to the fallback server.
SSL	Enable Backend Encryption	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.

Section	Setting	Description
Other	***Advanced*** > Maximum Connections	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 > Timeout	Use this option to override the default client & server timeouts in the Layer 7 advanced section. **Client Timeout** - The inactivity timeout applies when the client is expected to acknowledge or send data. **Real Server Timeout** - The inactivity timeout applies when the server is expected to acknowledge or send data.
	Advanced > Set X-Forwarded-For Header	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 > Force to HTTPS	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. **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: • 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. **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.

Section	Setting	Description
	Advanced > Use RIP name as Host Header	When set to 'Yes' the Host Header is set to the name allocated to the RIP.
	Advanced > Accept Proxy Protocol	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 the 'Enable Proxy Protocol' is enabled in your STunnel VIP. Note 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.
	Advanced > Send Proxy Protocol	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: • **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 > Enable Compression	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 > Set Source Address	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 > Enable HSTS	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 3 months or more.

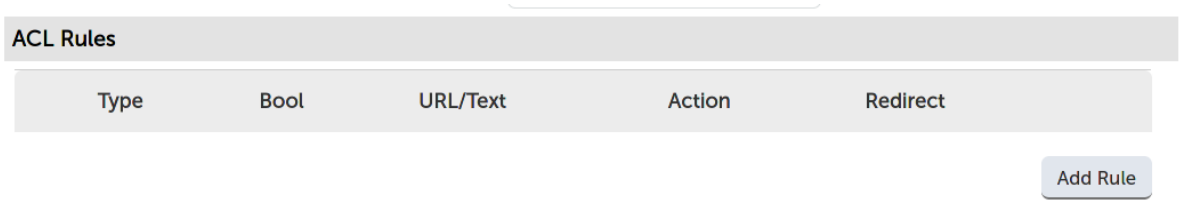
Section	Setting	Description
	Advanced > Tunnel Timeout	Timeout for the websocket protocol tunnel when no data is passed between client and server. Can be specified as s/m/h for seconds/minutes/hours.
	Advanced > Transparent Proxy	This will set the source address of the data stream leaving the load balancer 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: *Layer7 - Advanced Configuration > Transparent Proxy*.

Note

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).

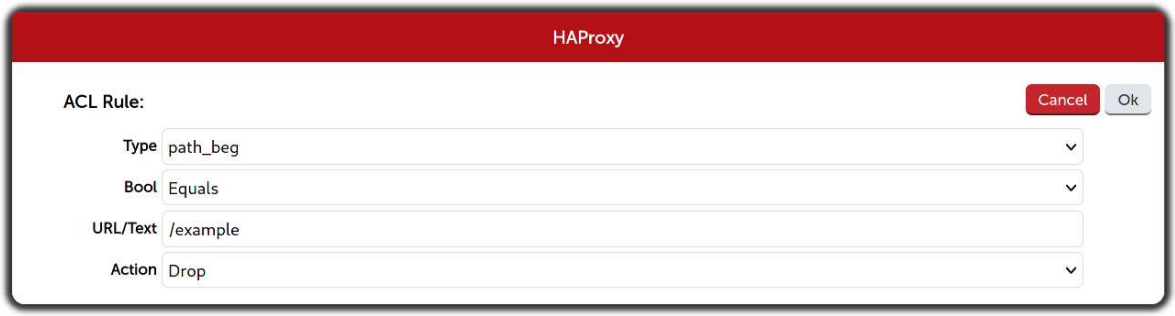
ACLs (aka Content Switching) and URL Rewriting

The WebUI supports the ability to create ACLs (access-control lists) which can be used to control and direct traffic based on a set of defined rules. This option can be accessed under the *ACL Rules* section by clicking the **Add Rule** button when modifying a VIP:

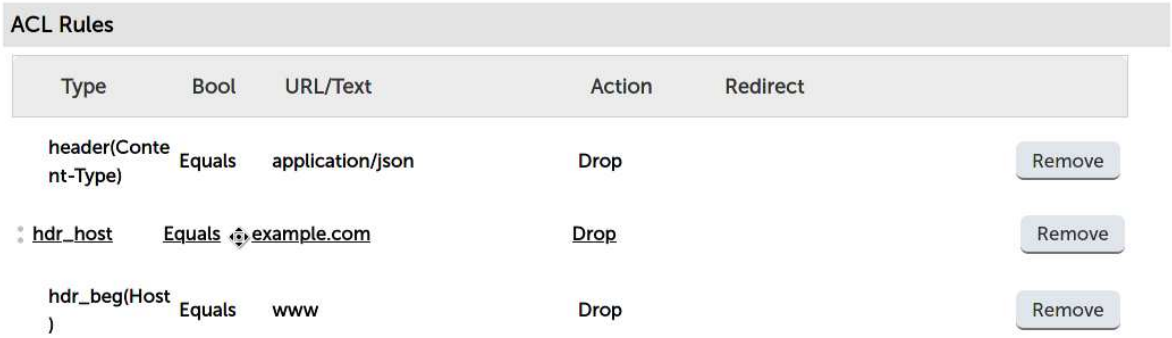


This brings up the ACLs pop-up menu:

Select the desired ACL type from the *Type* drop-down list, fill in the details as appropriate, and create the ACL by clicking the **Ok** button.



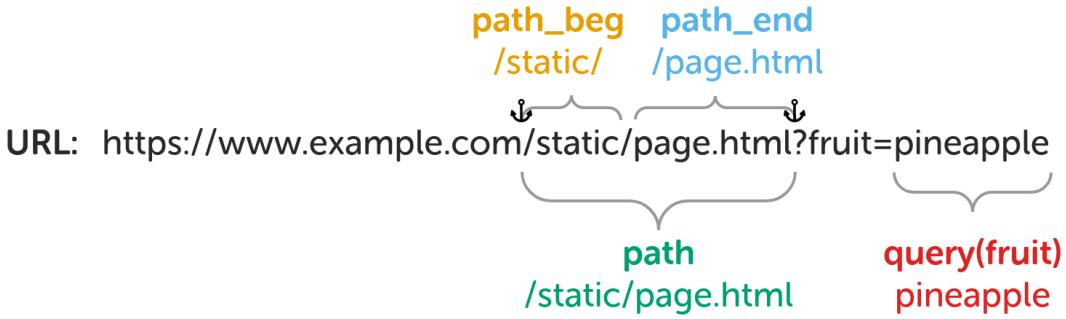
An ACL can be modified by clicking on it in the ACL list. ACLs can be reordered by clicking and dragging them in the ACL list:



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_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.

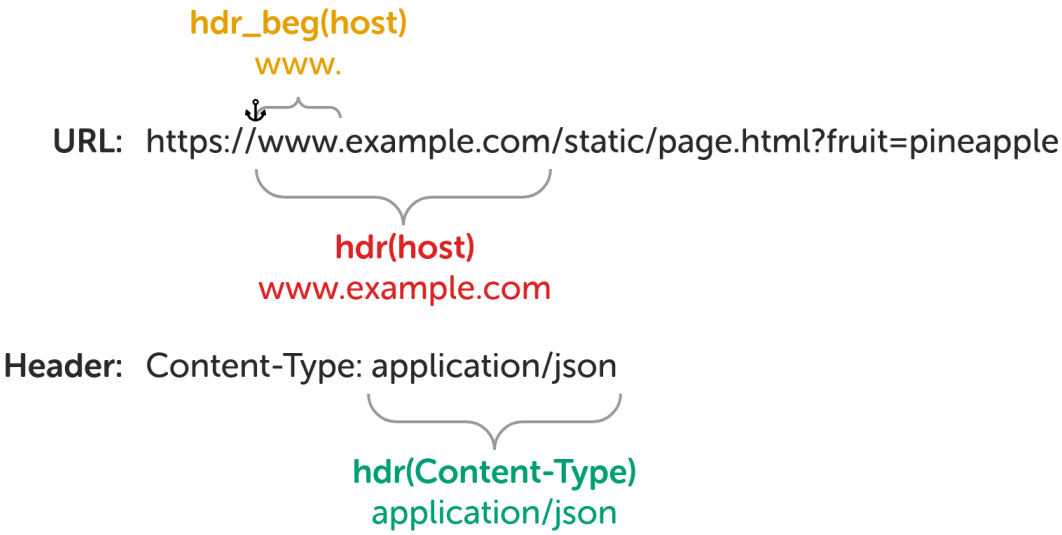
Query (aka url_param):

- 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_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 to write into the HAProxy configuration verbatim.
- **Freetype:** ACL configuration line to write into the HAProxy configuration file.

> 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	❌	✅
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 is 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: *User Server*, with the server name *apache_srv_dev*.

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 that defines the associated Real Servers. As a result, there is no floating IP address associated with the VIP. This kind of VIP is used exclusivly for ACLs where access is only needed from another VIP and not directly from clients over the network.

To create a Backend VIP:

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Virtual Services*.

2. Click **Add a New Virtual Service**.
3. Click [**Advanced**] and check (tick) 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.

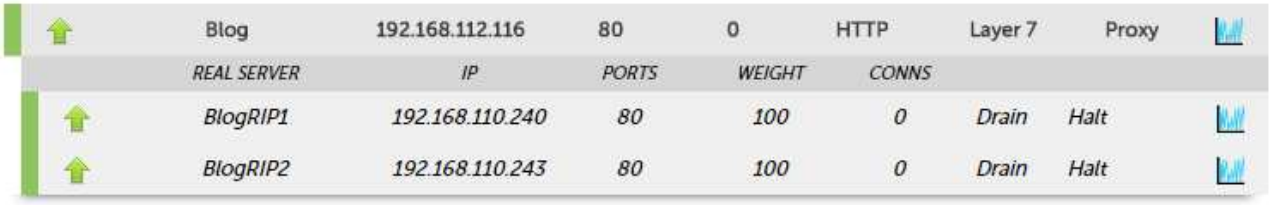
To create a manually defined backend:

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 2 Real Servers:



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 'reqirep' 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.

```
reqirep ^([^\ :]*)\ /static/(.*) \1\ /\2
```

Example 2

Replace any host name in the HTTP header with "www.mywebsite.com".

```
reqirep ^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

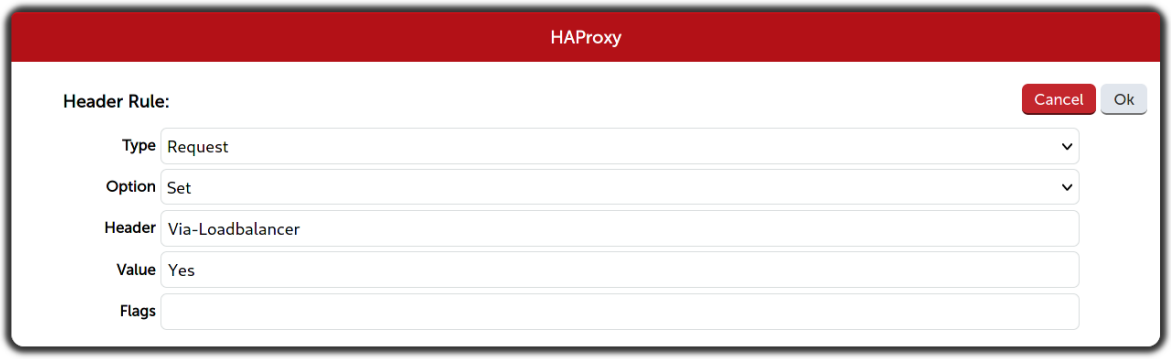
The "reqrep" keyword is strictly case-sensitive, while "repirep" is case insensitive. For more details on configuring manually defined layer 7 VIPs please refer to Configuring Manual Virtual Services.

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:

This brings up the headers pop-up menu:

Select the desired header type from the *Type* drop-down list, fill in the details as appropriate, and create the header rule by clicking the **Ok** button.



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:

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.

Option	Description
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 is possible to include dynamic information in header fields, for example the IP address of a request. For more details on this functionality please refer to the HAProxy online documentation available 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. For layer 7 VIPs port redirection is possible so the Real Server port field can be set to a different value to the VIP port. Real Servers in a Layer 7 configuration can be on any subnet in any network as long as 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.

3. Enter an appropriate *Label* (name) for the new Real Server.
4. Enter the required settings in the *Real Server IP Address* field and *Real Server Port* field.
5. 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).

6. Enable the *Enable Redirect* option if required. 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.

7. 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

The configuration options *Minimum Connections* and *Maximum Connections* are available when the Real Server is modified using **Modify** after the RIP has been created.

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 when creating and modifying VIPs & RIPs.

Configuring Manual Virtual Services

Often, the best approach is to first create a VIP and it's 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
Now, 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
Next, modify the VIP created in Step 1 and click **[Advanced]** in the *Virtual Service* section. Check (tick) the *Manual Configuration* checkbox, then click **Update**.

Step 5
Finally, navigate to: *Cluster Configuration > Layer 7 - Manual Configuration* and paste the configuration copied in Step 3 into the editor window, 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. V8.6 brings many enhancements that allow complex ACLs to be configured in the WebUI without the need for a manual configuration. For more information on configuring ACLs please refer to ACLs (aka Content Switching) and URL Rewriting.

The lines that need to be manually inserted to achieve this are:

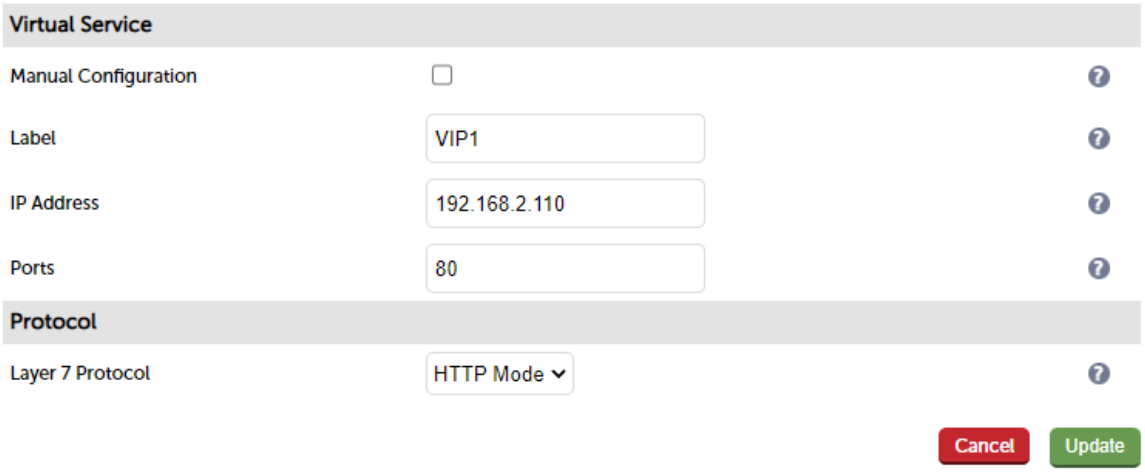
```
acl ACL-1 path_beg /staff/ (see note 1)
acl ACL-2 path_beg /staff (see note 1)
redirect location https://login.domain.com if ACL-1 or ACL-2 (see note 2)
```

Notes

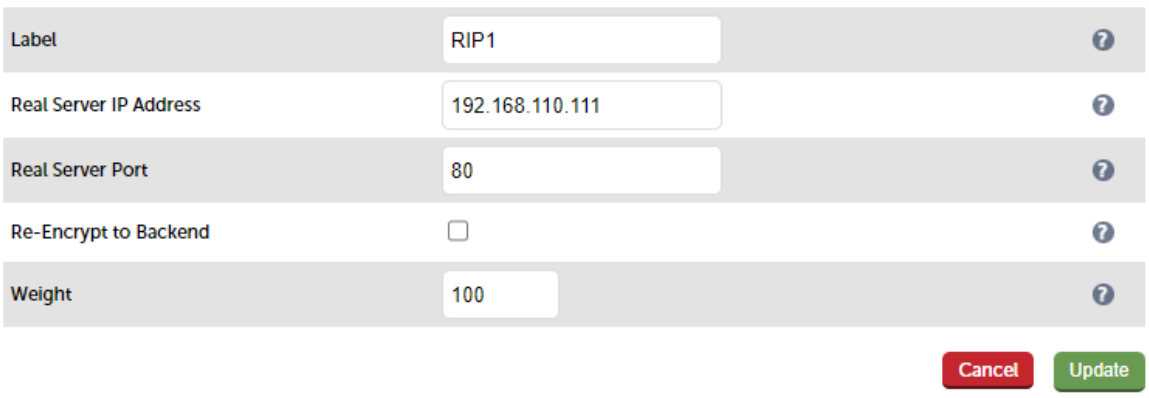
1. These lines configure 2 ACLs named **ACL-1** & **ACL-2** where the criteria for a match is that the URL starts with either **/staff/** or **/staff**.
2. This line causes a redirect to **https://login.domain.com** to occur when either ACL is matched.

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 un-checked, e.g.



2. Using the WebUI menu option: *Cluster Configuration > Layer 7 - Real Servers* define the associated RIPs in the normal way, e.g.



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
```

4. Using the WebUI menu option: *Cluster Configuration > Layer 7 - Virtual Services*, modify the VIP and check the *Manual Configuration* checkbox and click **Update**.
5. 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
```

6. Click **Update**.
7. Now reload HAProxy using the **Reload HAProxy** button in the blue *Commit Changes* box at the top of the screen.

Manual Config Example 2 - Load Balancing with URL Matching Using ACLs

Note

This example is for demonstration purposes only. V8.6 brings many enhancements that allow complex ACLs to be configured in the WebUI without the need for a manual configuration. For more information on configuring ACLs please refer to ACLs (aka Content Switching) and URL Rewriting.

In this example, the configuration is not modified from a standard config as in example 1 above, but created completely manually. To support URL matched load balancing the structure of the HAProxy configuration file must use the frontend/backend model as shown in the example below:

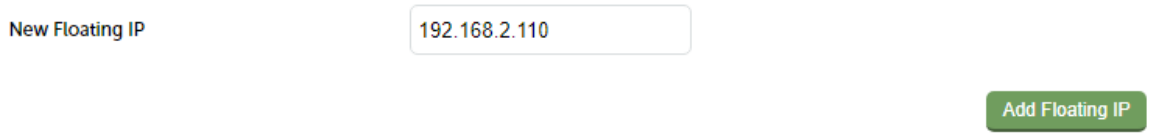
```
frontend f1
bind 192.168.2.110:80
acl ACL-1 path_beg /test1
acl ACL-2 path_beg /test2
use_backend b1 if ACL-1
use_backend b2 if ACL-2
default_backend b2
option httpclose

backend b1
cookie SERVERID insert nocache indirect
server s1 192.168.2.111:80 weight 100 cookie s1 check
server s2 192.168.2.112:80 weight 100 cookie s2 check

backend b2
cookie SERVERID insert nocache indirect
server s3 192.168.2.113:80 weight 100 cookie s3 check
server s4 192.168.2.114:80 weight 100 cookie s4 check
```

Notes

1. **ACL-1** & **ACL-2** are the names of the ACLs.
2. **path_beg** matches the beginning of the path to a certain value, in this case /test1 & /test2 and then directs requests to the appropriate backend, either B1 or B2.

Configuration Steps:

1. Using the WebUI menu option: *Cluster Configuration > Floating IPs* , add a floating IP for the new VIP, in this example 192.168.2.110 is added to match the IP address required:



2. Click **Add Floating IP**.
3. Select the WebUI menu option: *Cluster Configuration > Layer 7 - Manual Configuration* and define the required VIP/RIP settings in the editor window:

```
frontend F1
bind 192.168.2.110:80
acl ACL-1 path_beg /test1
acl ACL-2 path_beg /test2
use_backend B1 if ACL-1
use_backend B2 if ACL-2
default_backend B2
option httpclose

backend B1
cookie SERVERID insert nocache indirect
server s1 192.168.2.111:80 weight 100 cookie s1 check
server s2 192.168.2.112:80 weight 100 cookie s2 check

backend B2
cookie SERVERID insert nocache indirect
server s3 192.168.2.113:80 weight 100 cookie s3 check
server s4 192.168.2.114:80 weight 100 cookie s4 check
```

4. Click **Update**.
5. Now reload HAProxy using the **Reload HAProxy** button in the blue *Commit Changes* box at the top of the screen or by using the WebUI menu option: *Maintenance > Restart Services*.

Note

This example uses the Frontend/Backend structure to define the Layer 7 Virtual Service. When using this structure, the related Virtual Service cannot be displayed in the System Overview so there is no need to define a matching VIP in this case.

These are fairly simple examples to show the principle of creating manual layer 7 configs and also creating and using ACLs. For more information on configuring ACLs please refer to ACLs (aka Content Switching) and URL Rewriting. Please also refer to the HAProxy manual available here.

Note

Don't hesitate to contact support@loadbalancer.org to discuss any specific ACL or other custom configuration requirements you may have.

HAProxy Error Codes

For reference, HAProxy's own error codes are as follows:

Code	When/Reason
200	access to stats, 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.
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.
408	when the request timeout strikes before the request is complete.
500	when HAProxy encounters an unrecoverable internal error, such as a memory allocation failure, which should never happen.

Code	When/Reason
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 text based manual or the HTML based manual.

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 by default 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 Ethernet 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 Ethernet interface is used.

Enabling Transparency

The load balancer can provide the actual client IP address to the Real Servers in 2 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 about XFF headers please click here, for more information about Proxy Protocol Headers please click here.

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.

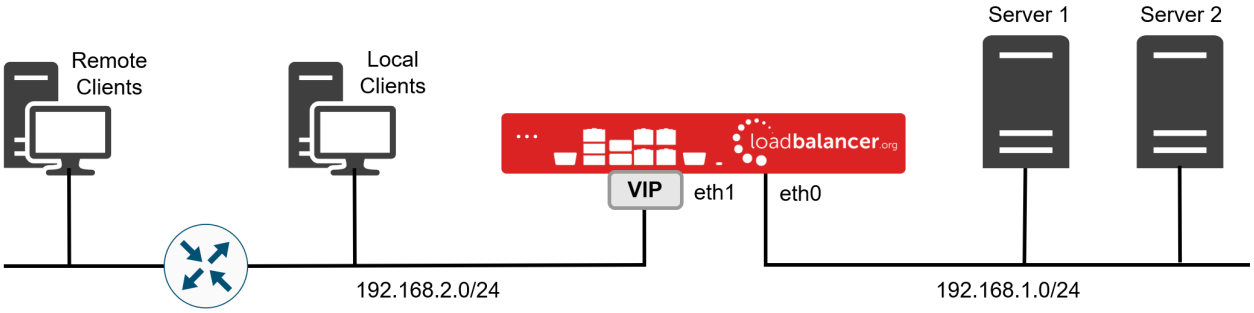
HAProxy - To configure HAProxy to send Proxy Protocol Headers, the layer 7 Virtual Service drop-down *Send Proxy Protocol* must be set to the required header version/type. To configure HAProxy to receive Proxy Protocol Headers, 2 methods can be used:

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.
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.

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 - Two-arm Deployments

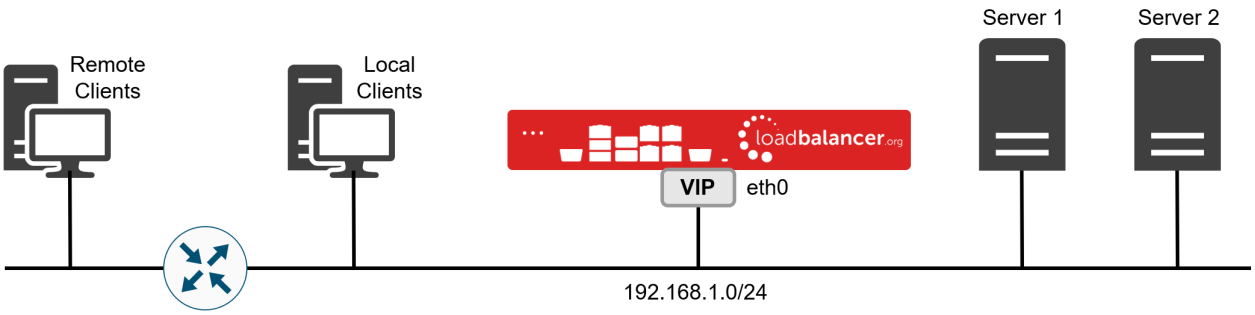


- Here, 2 subnets are used. The VIP is located in one subnet and the load balanced Real Servers are located in the other. The load balancer requires 2 interfaces, one in each subnet.

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

- The default gateway on the Real Servers must be an IP address on the load balancer.
- Clients can be located in the same subnet as the VIP or any remote subnet provided they can route to the VIP.

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

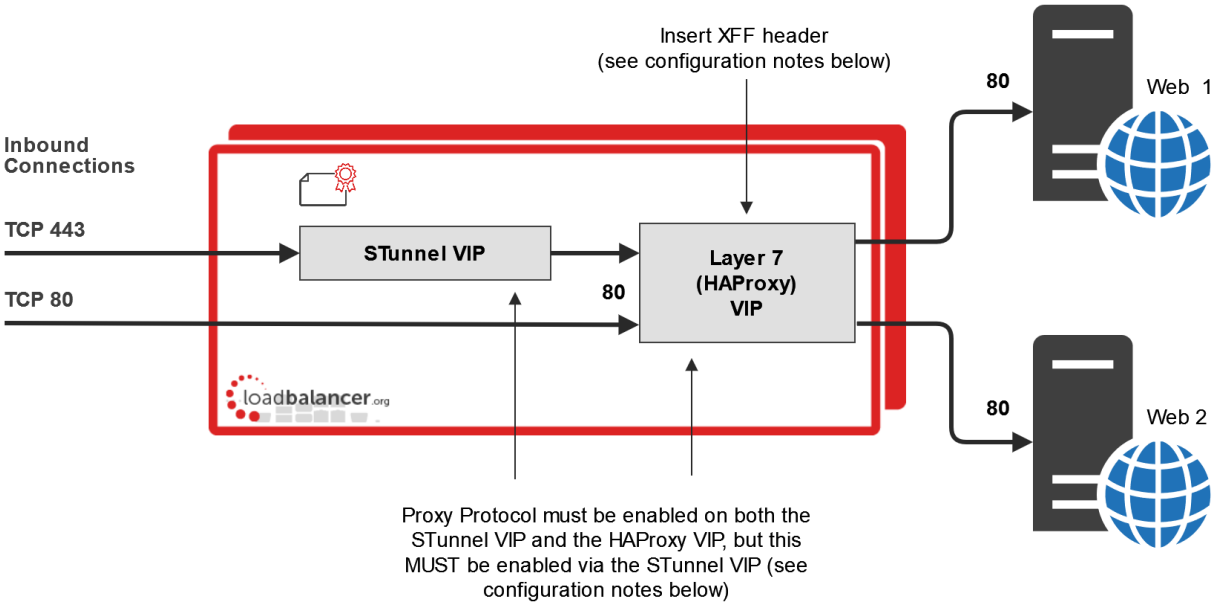
For an HA clustered pair, a floating IP should be added to the load balancer and used as the Real Server's default gateway. This ensures that the IP address can 'float' (move) between Primary and Secondary appliances.

- 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.

Configuration Examples

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.

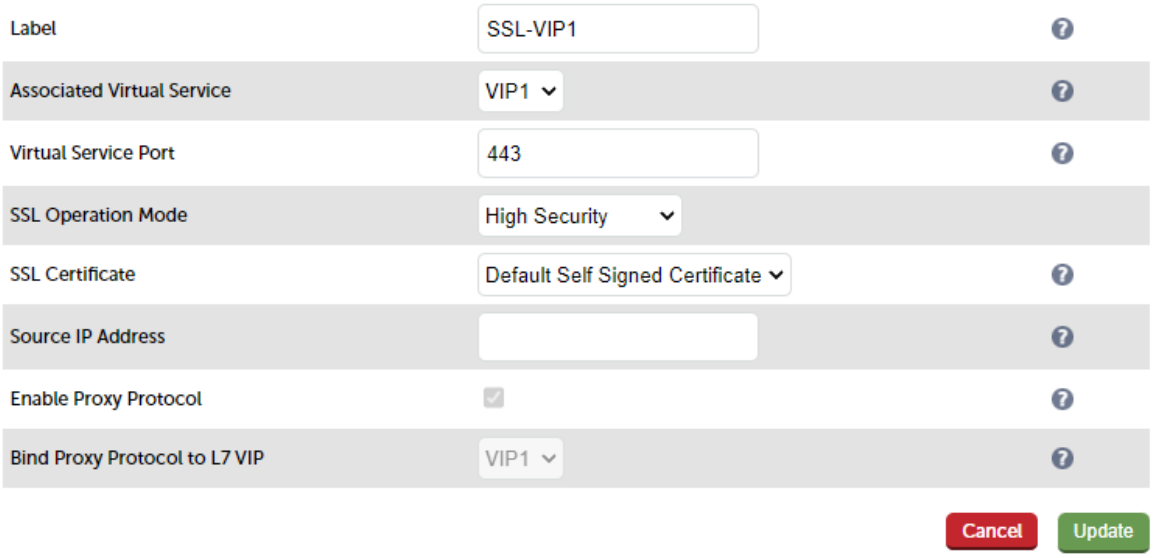


Configuration Notes

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:

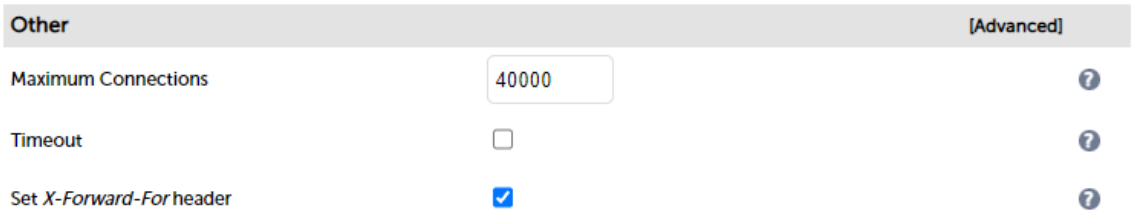


The STunnel VIP option *Associated Virtual Service* must be set to the backend HAProxy VIP where STunnel will forward its traffic. Then, both the STunnel VIP and the associated HAProxy VIP will be configured automatically.

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:

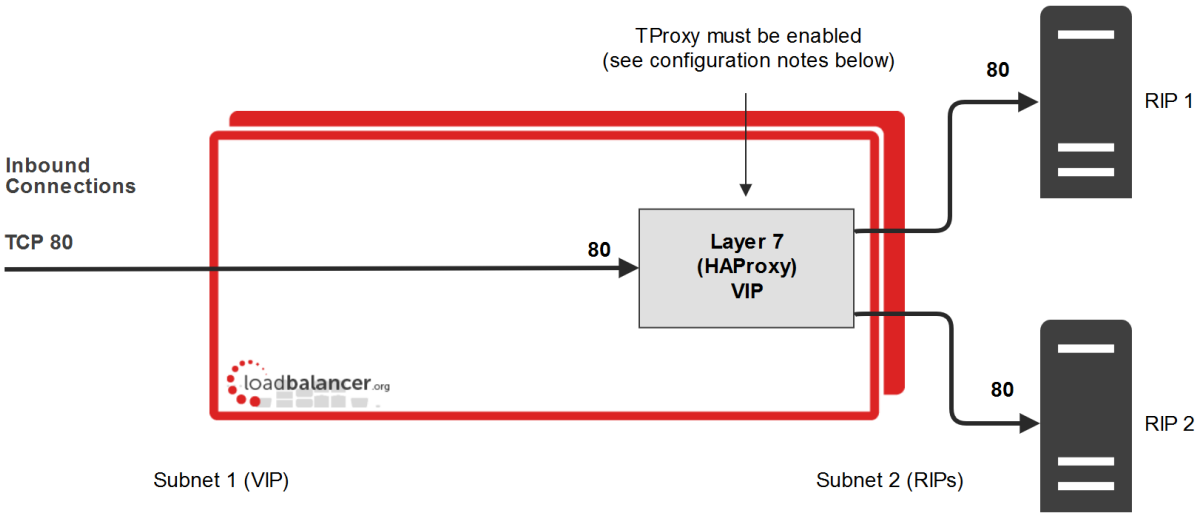


X-Forwarded-For Headers must be enabled for HAProxy (this is the default setting).

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.

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 clients IP address.

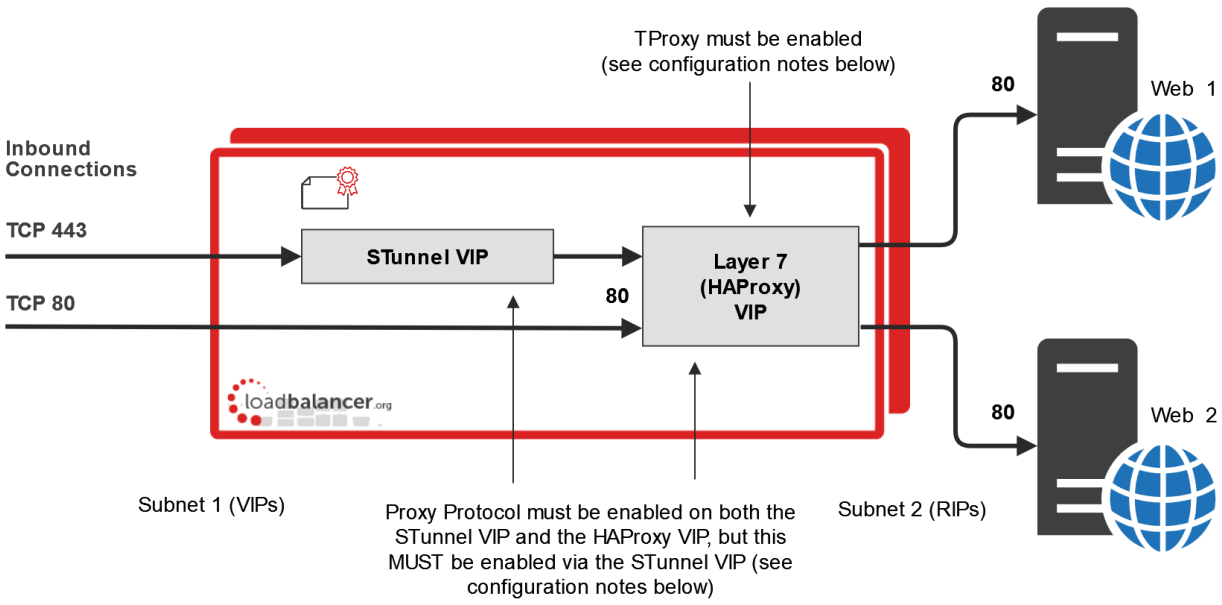


Configuration Notes & Topology Requirements

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.

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 clients IP address.



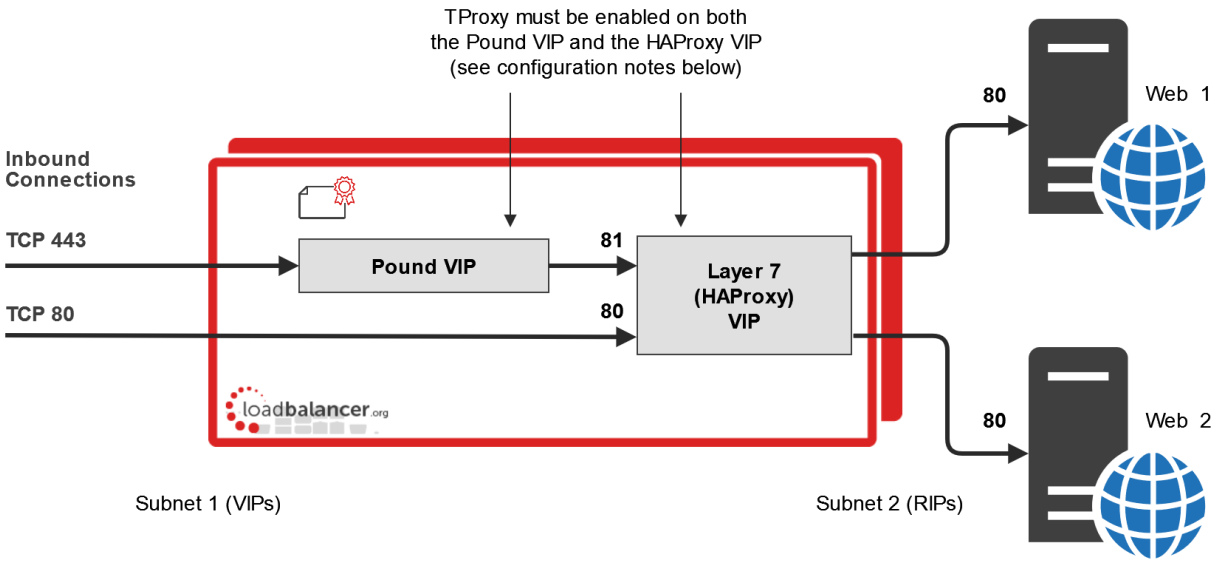
Configuration Notes & Topology Requirements

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.

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 clients 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 2 ports - e.g. 80 & 81, then use port 80 for client connections on HTTP and port 81 for the Pound backend.
4. When defining 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 2 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

It's now possible to configure each virtual service as read-only. The manual configuration can then be created using the WebUI option: *Cluster Configuration > Layer 7 - Manual Configuration*. For More information on configuring manual layer VIPs please refer to Configuring Manual Virtual Services.

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 '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 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.

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.

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 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.

Floating IPs

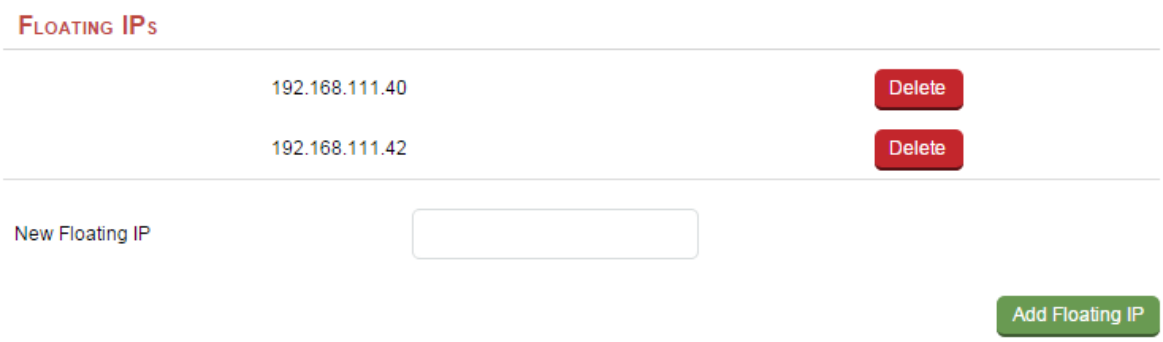
In order for the load balancer to function, the unit must physically own the Virtual IP address that the clients are accessing before they get re-directed to a Real Server in the cluster. When new layer 4 or layer 7 Virtual Services (VIPs) are created, Floating IPs are added automatically and can be viewed using the WebUI menu option: *Cluster Configuration > Floating IPs*.

It's also possible to manually define 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.

The Floating IPs are controlled by heartbeat to ensure that only one of the load balancer appliance's (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*.



2. Specify the new floating IP.
3. 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	Floating IPs are not deleted automatically when Virtual Services are removed or the IP address is changed, this must be done manually.

SSL Termination

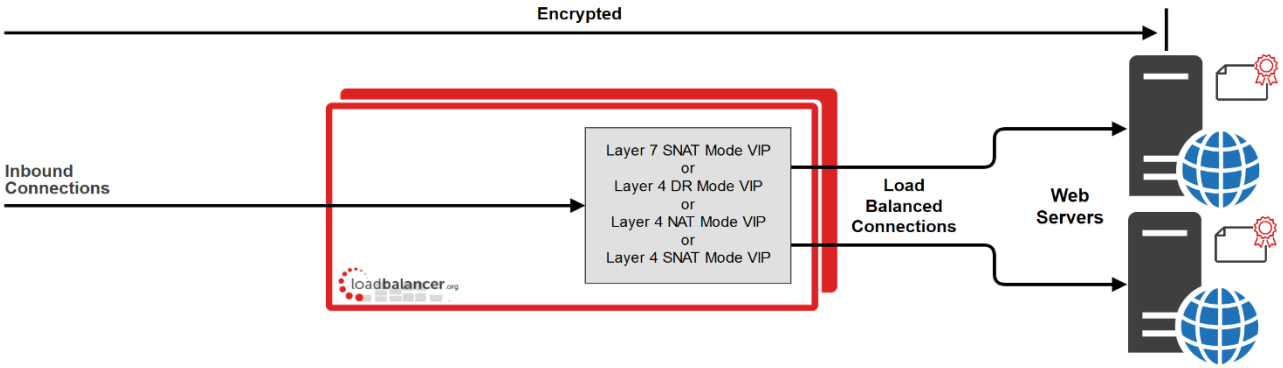
Concepts

SSL termination can be handled in the following ways:

1. On the Real Servers (recommended) - aka ***SSL Pass-through***.
2. On the load balancer - aka ***SSL Offloading***.
3. 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 Pass-through)



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

1. 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.
2. 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

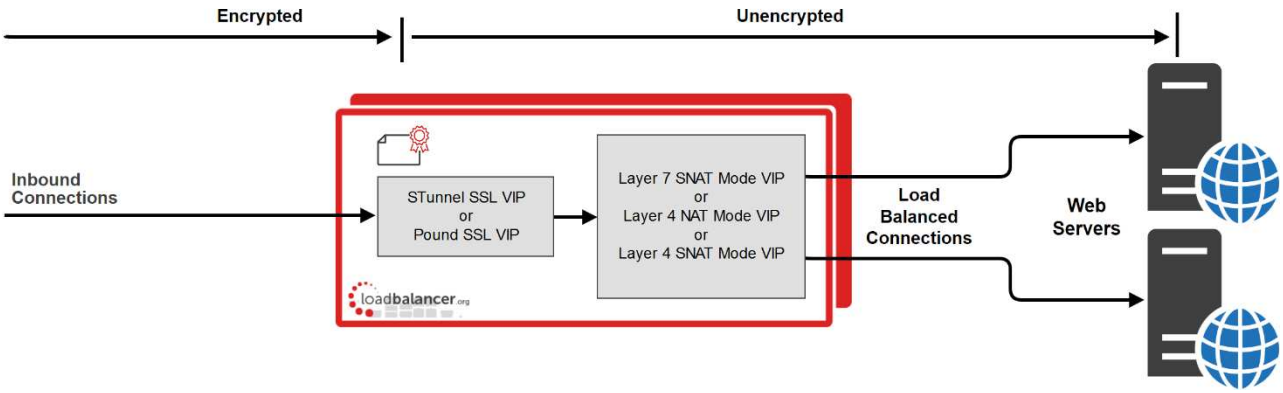
A fairly common configuration is to include port 80 in the VIPs definition and also enable persistence. This ensures that both HTTP and HTTPS requests from a particular client are always sent to the same Real Server as shown below:

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.



In this case an STunnel or Pound SSL Virtual Service is defined on the appliance and an SSL certificate is uploaded and associated to the Virtual Service. Data is encrypted from the client to the load balancer, but is unencrypted from the load balancer to the backend servers as shown above. If you require SSL bridging where the data is re-encrypted from the load balancer to the backend servers, please refer to SSL Termination on the Load Balancer with Re-encryption (SSL Bridging).

Notes

1. By default, a self-signed certificate is used for the new Pound/STunnel VIP. Certificates can be created or uploaded as described in the section below. The self-signed certificate can be regenerated if needed using the WebUI menu option: *SSL Certificate* and clicking the **Regenerate Default Self Signed Certificate** button.
2. The backend for the STunnel / Pound VIP can be either a Layer 7 SNAT mode VIP or a Layer 4 NAT or SNAT mode VIP. Layer 4 DR mode cannot be used since Pound & STunnel act as a proxy, and the real servers see requests with a source IP address of the VIP. However, since the Real Servers believe that they own the VIP (due to the loopback adapter configured to handle the ***ARP Problem***) they are unable to reply to Pound.
3. If a layer 7 VIP is used as the backend for the STunnel or Pound VIP, it's possible to use cookie based persistence as well as other layer 7 techniques to control traffic flow to the Real Servers.

Certificates

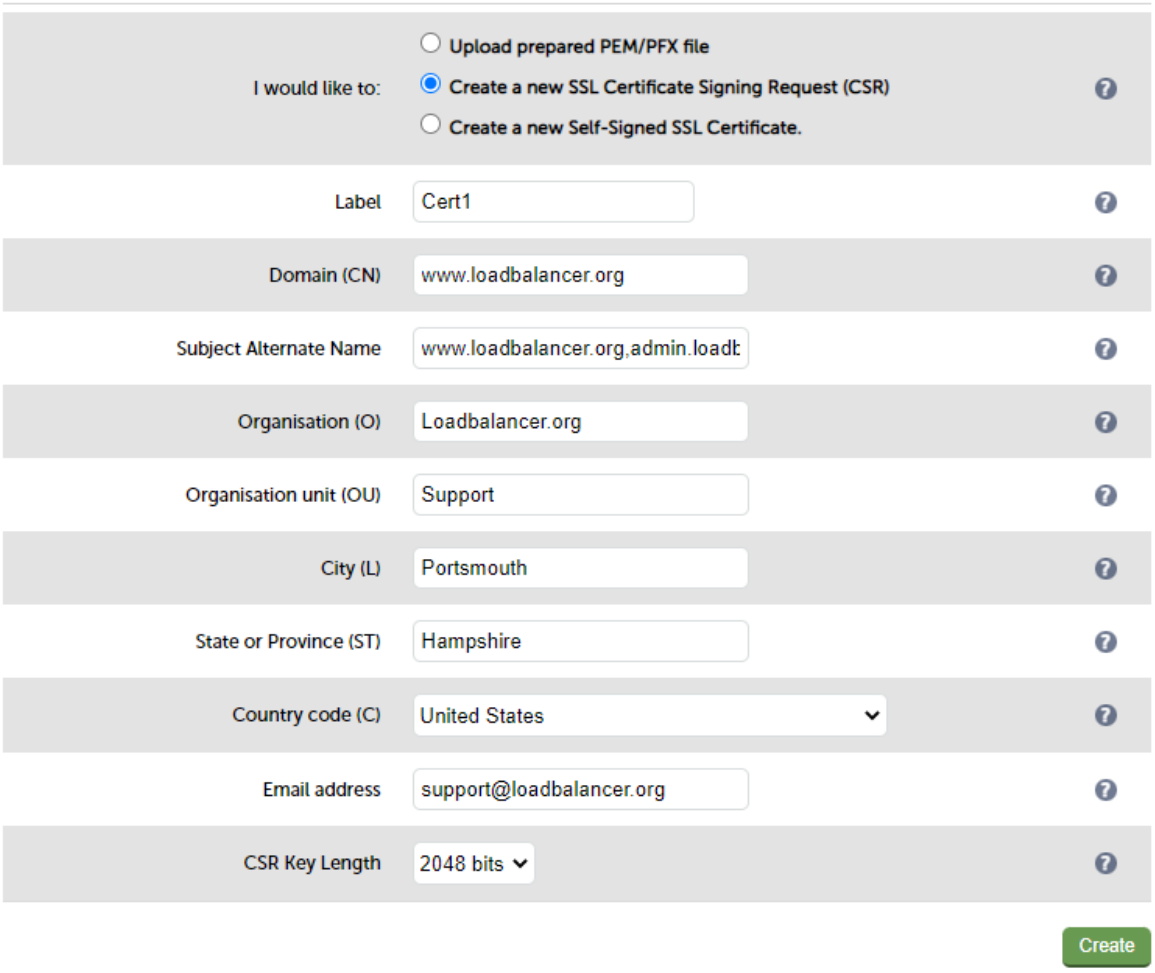
If you already have an SSL certificate in either PFX or PEM file format, this can be uploaded to the Load balancer using the certificate upload option as explained in Uploading Certificates. Alternatively, you can create a Certificate Signing Request (CSR) and send this to your CA to create a new certificate, 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)*.



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:

> **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 unit.

> **Note** It's important to backup all your certificates. This can be done via the WebUI from *Maintenance > Backup & Restore > Download SSL Certificates*.

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:

- 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-----
```

Converting between certificate formats

In some circumstances it may be required to manually convert certificates between formats. In these cases OpenSSL can be used. This is usually included by default in Linux distributions. For Windows, it can be downloaded here.

At this URL you can download either the light or full version of OpenSSL. Once installed, you'll have an OpenSSL directory located on your filesystem (default location 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 detailed below.

Converting PFX certificates to PEM format

1) Using OpenSSL on Windows:

```
openssl pkcs12 -in file.pfx -nodes -out file.pem
```

2) Using the Appliance/Linux:

```
openssl pkcs12 -in file.pfx -nodes -out file.pem
```

Converting .cer certificates to PEM format

1) Using OpenSSL on Windows:

```
openssl x509 -in file.cer -inform DER -out file.pem -outform PEM
```

2) Using the Appliance/Linux:

```
openssl x509 -in file.cer -inform DER -out filen.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 OpenSSL command either on the load balancer or another machine with OpenSSL installed:

```
openssl rsa -in encrypted-server.key -out unencrypted-server.key
```

Let's Encrypt

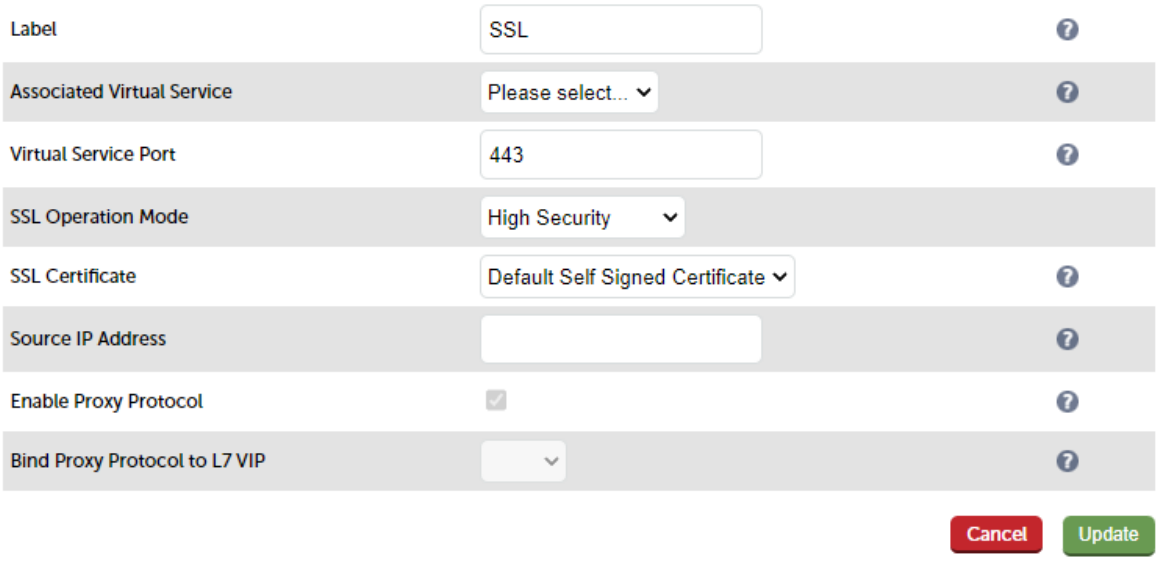
Lets Encrypt is s zero cost Certificate Authority for HTTPS encryption, now trusted by all major root programs, including Google, Microsoft, Apple, Mozilla and Oracle. Used in conjunction with freely available tools it provides automatic enrollment/renewal, simple cert creation, negating validation emails and manual configuration.

For much more information, please refer to our Let's encrypt introductory blog and also the follow up blog that details the new **lb-letsencrypt.sh** script and how to use it.

Creating a SSL Termination

To add a SSL Termination:

1. Using the WebUI, navigate to: *Cluster Configuration > SSL Termination*.
2. Click **Add a new Virtual Service**.



3. Enter an appropriate *Label* (name) for the new Virtual Service.

> Note | The label will be auto configured based on the *Associated Virtual Service* selected. This can be edited if required.

4. Select the *Associated Virtual Service* where you want to forward the unencrypted STunnel traffic or select **None** to manually configure these settings (more information on manually configuring these settings is available here).

> Note | From v8.5.4 the same Virtual Service can be associated with multiple SSL terminations.

5. Enter the required port in the *Virtual Service Port* field - typically **443**.
6. Select the required *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 (more information on manually configuring these settings is available here).

The following STunnel settings are auto-configured for each SSL Operation Mode:

STunnel Setting	High Security	FIPS Compliant	High Compatibility
Delay DNS Lookups	Y	Y	Y
Disable SSLv3 Ciphers	Y	Y	Y
Disable TLSv1.0 Ciphers	Y	Y	N
Disable TLSv1.1 Ciphers	Y	Y	N
Disable TLSv1.2 Ciphers	N	N	N
Disable TLSv1.3 Ciphers	N	Y	Y
Honor Cipher Order	Y	Y	Y
Don't Insert Empty Fragments	Y	Y	Y
Disable SSL Renegotiation	Y	Y	Y

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-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
```

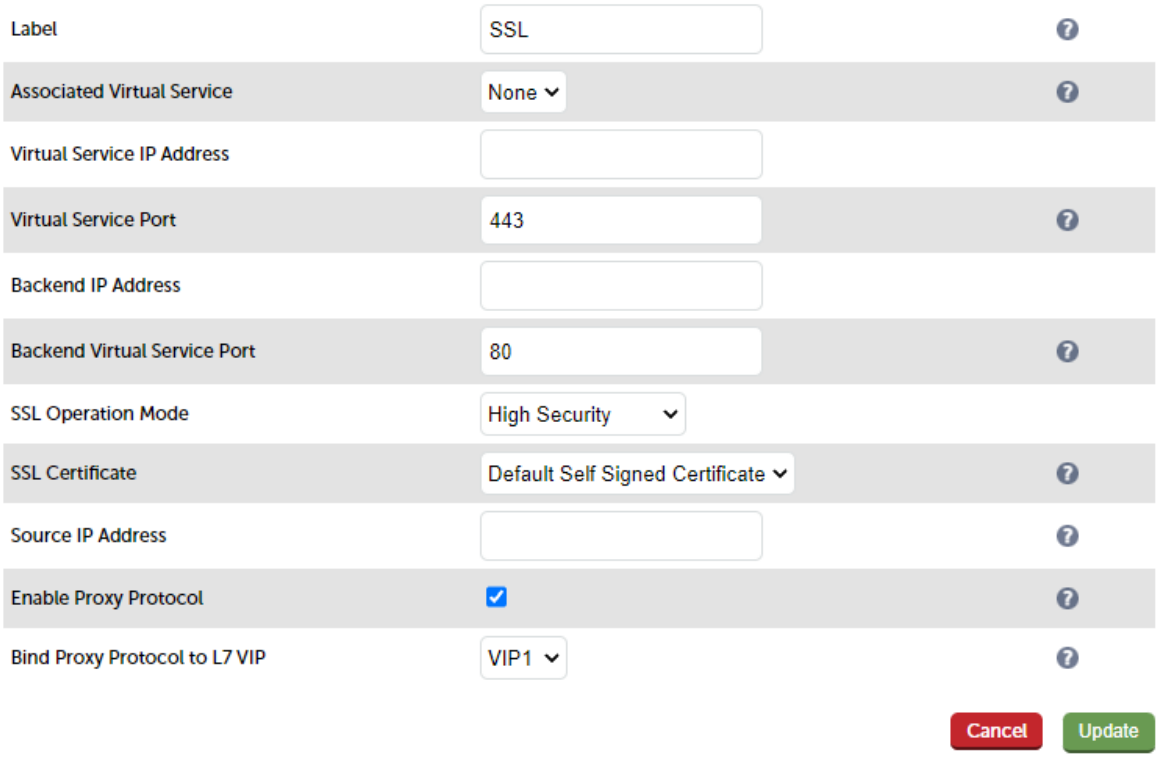
7. Select the required certificate from the drop-down.

> **Note** If you have not added any certificates at this point, a self signed cert will be used.

8. Click **Update**.

Associated Virtual Service - Manual Mode

If you set *Associated Virtual Service* to **None**, the following settings can be configured manually:



1. Enter the required *Label* (name) for the Virtual Service.
2. Enter the required *Virtual Service IP Address*.
3. Enter the required *Virtual Service Port* - typically 443.
4. 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 HAProxy VIP or a Layer 4 NAT / SNAT mode VIP. Unencrypted traffic will be sent here for load balancing.

> **Note** Layer 4 DR mode cannot be used since STunnel acts as a proxy, and the Real Servers see requests with a source IP address of the Virtual Service. However since the Real Servers believe that they own the Virtual IP (due to the Loopback Adapter configured to handle the ***ARP Problem***) they are unable to reply to STunnel.

5. Enter the required *backend Virtual Service Port*.
6. Select the required *SSL Operation Mode* (more information on the various modes is available here).
7. Select the required *SSL Certificate*.

8. Set the required *Source IP Address* - by default the *Virtual Service IP Address* is used but this can be changed if required.
9. 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.
10. Configure *Bind Proxy Protocol to L7 VIP* - This option is available 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 clients 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 configurations are not included in the drop-down.
11. Click **Update**.

SSL Operation Mode - Custom Mode

If you set the *SSL Operating Mode* to **Custom**, you'll be able to configure all settings manually.

1. Select the required *SSL Certificate*.
2. Define the list of accepted ciphers using the *Ciphers to use* field. By default the cipher is set to:

   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

   This can be modified as required, or the field can be cleared (blank) to allow all available ciphers (not recommended)

3. Configure *Disable SSLv3 Ciphers* - When ticked this option disables all SSLv3 Ciphers.
4. Configure *Disable TLSv1.0 Ciphers* - When ticked this option disables all TLSv1.0 Ciphers.
5. Configure *Disable TLSv1.1 Ciphers* - When ticked this option disables all TLSv1.1 Ciphers.
6. Configure *Disable TLSv1.2 Ciphers* - When ticked this option disables all TLSv1.2 Ciphers.
7. Configure *Disable TLSv1.3 Ciphers* - When ticked this option disables all TLSv1.3 Ciphers.
8. Set *SSL Terminator* to **STunnel** or **Pound**.

If STunnel is selected (Recommended) :

1. Configure *Do not Insert Empty Fragments* - This option needs to be enabled (checked) to ensure mitigation of both the BEAST and CRIME MITM attacks. It is also required for PCI Testing.
2. Configure *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).
3. Configure *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.
4. Configure *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.

5. Configure *Time to Close* - Configure the global client response timeout in seconds. This setting should not require changing.
6. Configure *Set Source Address* - By default the Virtual Service IP Address will be used as the STunnel Source Address. However, if you have a large amount of traffic this may cause an issue and you can change the source IP Address to allow for extra capacity.
7. 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.
8. Configure *Bind Proxy Protocol to L7 VIP* - This option is available 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 clients 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 configurations are not included in the drop-down.
9. Click **Update** to create the STunnel VIP.

If Pound is selected:

1. Configure *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
2. If required, configure *Header Field Name* & *Header Field Value* to define a custom Pound header.
3. Configure *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.
4. Configure *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.
5. Configure *Client Cipher Renegotiation* - Sets whether the client is allowed to renegotiate the cipher order:
   - No Client Renegotiation - no client renegotiation will be honored
   - Secure Renegotiation - secure renegotiation will be honored
   - Insecure Renegotiation - insecure renegotiation will be honored
6. Click **Update** to create the Pound VIP.

Once the SSL Termination is created, the associated VIP will be displayed with a padlock symbol in the system overview as shown below:

Server Name Indication (SNI)

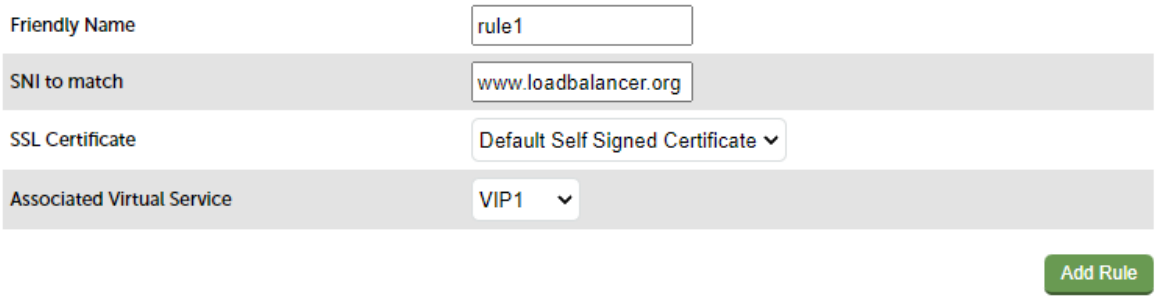
Server Name Indication (SNI) is an extension to the TLS protocol which allows a client to indicate which hostname it is attempting to connect to at the start of the handshaking process. This allows the load balancer to present multiple secure websites on the same IP address and port, but with different certificates. SNI rules are associated with an STunnel VIP to define which certificate is presented and which backend traffic should be forwarded to. The following section provides more details on configuring SNI.

Configuring Server Name Indication (SNI) Rules

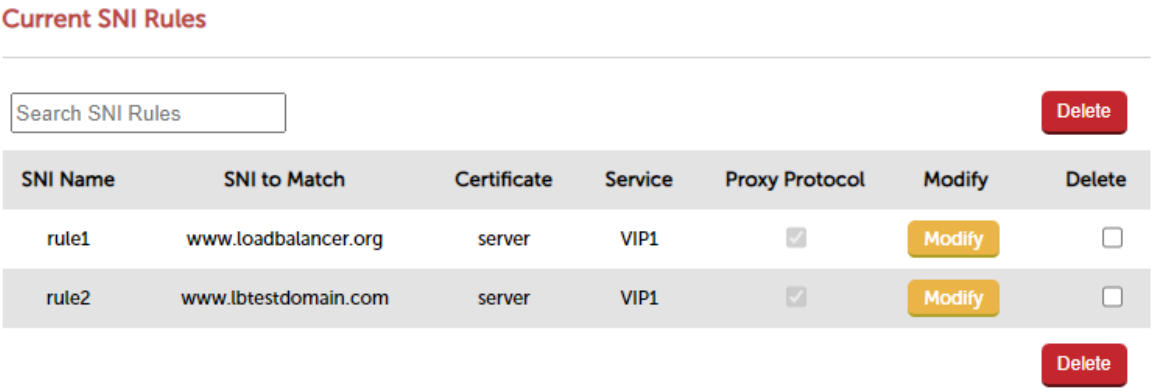
SNI matching allows you to send traffic to different backend Virtual Services based on the FQDN requested. SNI rules must be configured after the STunnel VIP is created.

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**.

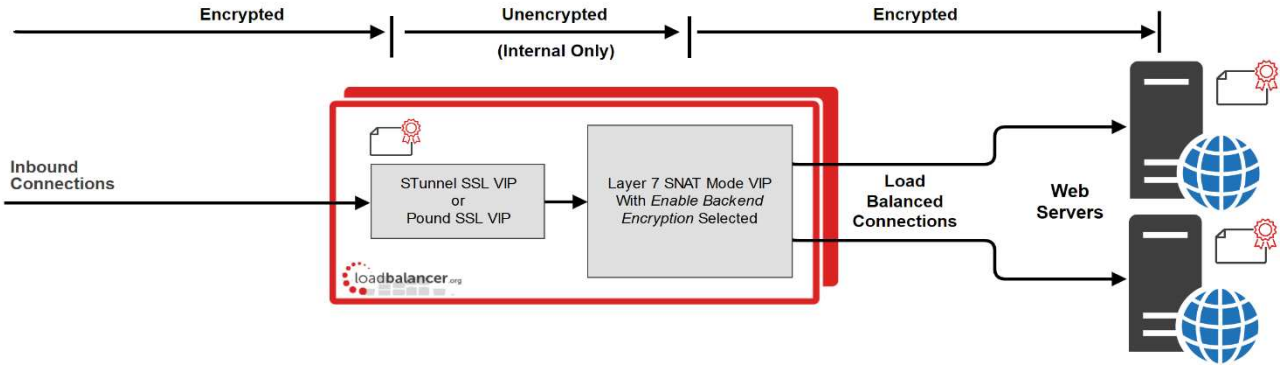


4. Enter a suitable *Friendly Name* for the new Rule, e.g. **rule1**.
5. Enter the required *SNI to Match* e.g. **www.loadbalancer.org**.
6. Select the required *SSL Certificate*.
7. Use the *Associated Virtual Service* drop-down to select the required Virtual Service, or select **Custom** and enter the required *Backend IP Address* and *Backend Virtual Service Port* and configure the *Enable Proxy Protocol* checkbox according to your requirements.
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:



11. Modify or Delete SNI rules using the buttons provided.
12. To apply the new settings, restart STunnel using the **Reload STunnel** button 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:

SSL Termination on the Load Balancer with Re-encryption (SSL Bridging)



In this case, an STunnel or Pound SSL Virtual Service is defined on the appliance and an SSL certificate is uploaded and associated with the Virtual Service. Data is encrypted from the client to the load balancer and is also encrypted from the load balancer to the backend servers as shown above.

Notes

1. This is similar to SSL Offload, the only difference is that the connection from the load balancer to the Real Servers is encrypted using the certificate located on the real server, this could be a self-signed certificate since no client connections are terminated here, only at the STunnel or Pound VIP.
2. This mode can be enabled for the entire VIP and all associated Real Servers using the VIP option *Enable Backend encryption* or per Real Server using the *Re-Encrypt to Backend* option as detailed below.

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.

To enable re-encryption at the Virtual Service level:

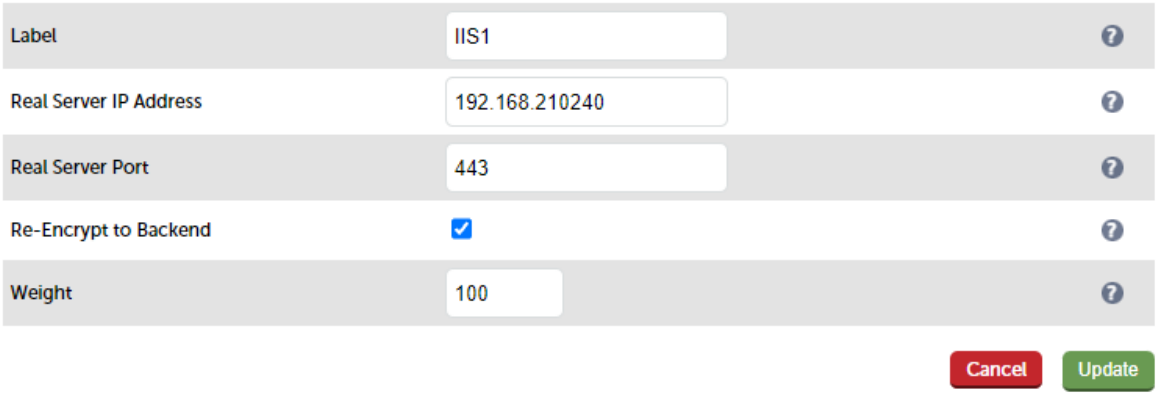
1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Virtual Servers*.
2. Click **Modify** next to relevant Virtual Service.

3. Scroll down to the *SSL* section and check (enable) the *Enable Backend Encryption* checkbox.
4. Click **Update** - you'll be asked if you want to apply the setting to all existing backend servers - click **OK** or **Cancel** as required.

Note | The new setting will automatically apply to all new Real Servers added.

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.



3. Check (enable) 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

One 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 2 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

Debug Level - Option to set the debugging level for all STunnel Services. The Debug Level is a one of the syslog level names or numbers emergency (0), Alert (1), Critical (2), err (3), Warning (4), Noti ce (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.

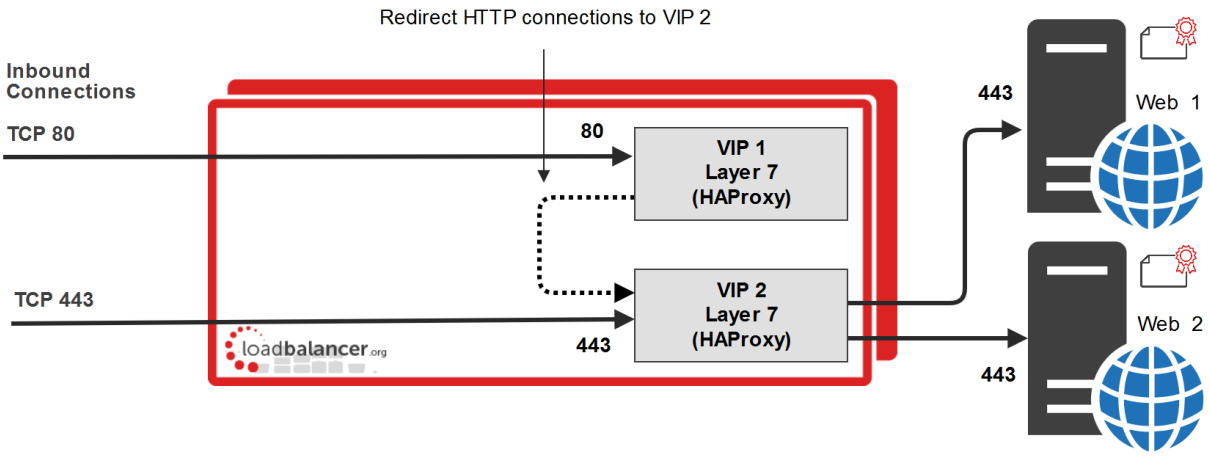
Enable FIPS 140-2 Mode - FIPS (Federal Information Processing Standards) are a set of standards that describe document processing, encryption algorithms and other information technology standards for use within non-military government agencies and by government contractors and vendors who work with the agencies. Check to enable FIPS 140-2 mode for STunnel.

HTTP to HTTPS Redirection

The appliance supports the ability to force HTTP to HTTPS redirection. This can be achieved 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 2 VIPs.



VIP 1 & VIP 2 are configured on the same IP address for HTTP/HTTPS client connections

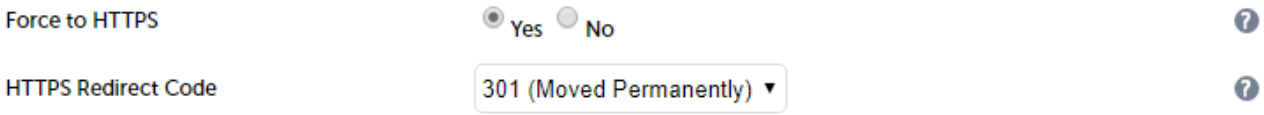
- **VIP 1** - This is a layer 7 HTTP mode VIP that listens on port 80 and redirects all connections to VIP2. It has the option *Force to HTTPS* enabled which redirects the HTTP client connections (see below).

> **Note** VIP1 will show purple/green in the System Overview. This occurs once *Force to HTTPS* is enabled (see below). This VIP does not need any Real Servers to be configured.

- **VIP 2** - This is a layer 7 TCP mode VIP that listens on port 443 and load balances connections between Real Servers Web 1 & Web 2.

VIP 1 Redirect Configuration

Click **Modify** next to the VIP, enable the *Other (Advanced)* > *Force to HTTPS* option, and set the redirect code as required as shown in the example below:

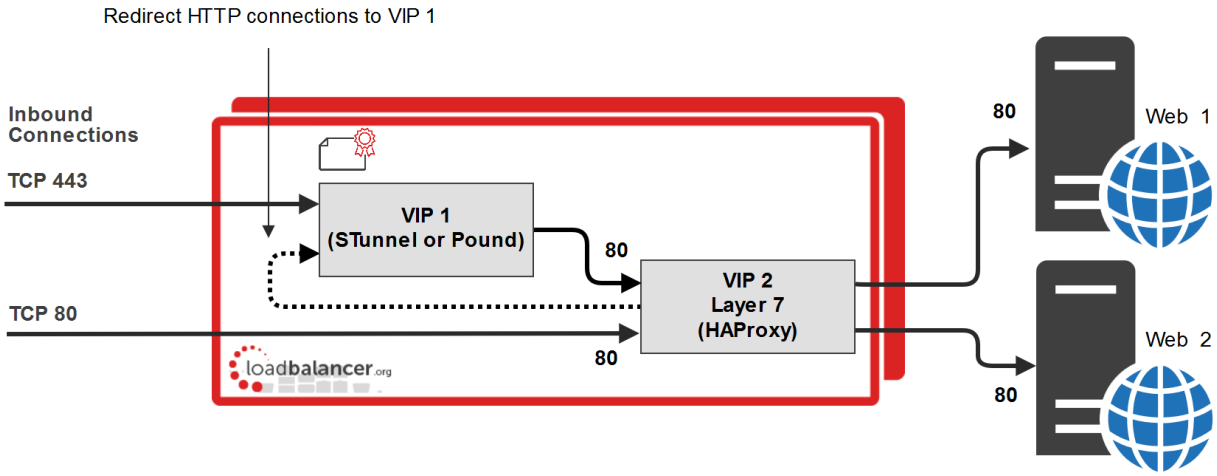


> **Note** The *Force to HTTPS* option is only available when the VIP is in HTTP mode.

> **Note** It's not possible to enable TProxy when using this configuration.

When Terminating SSL on the Load Balancer

This method requires 2 VIPs.

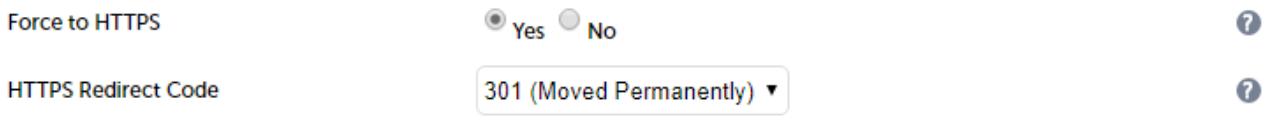


VIP 1 & VIP 2 are configured on the same IP address for HTTP/HTTPS client connections

- **VIP 1** - This is a Pound or STunnel VIP that listens on port 443, terminates the SSL connection and then forwards the decrypted HTTP connections to VIP2 on port 80.
- **VIP 2** - This is a layer 7 HTTP mode VIP that listens on port 80 and load balances connections between Real Servers Web 1 and Web 2. It has the option *Force to HTTPS* enabled which redirects the HTTP client connections (see below).

VIP 2 Redirect Configuration

Click **Modify** next to the VIP, enable the *Other (Advanced)* > *Force to HTTPS* option, and set the redirect code as required as shown in the example below:



Note	The *Force to HTTPS* option is only available when the VIP is in HTTP mode.
Note	It's not possible to enable TProxy when using this configuration.
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 modify the weight (amount of traffic) of each server by gathering data from either a custom agent or an HTTP server. For layer 4 VIPs the feedback method can be set to either agent or HTTP, for Layer 7 VIPs, only the agent method is supported.

A telnet to port 3333 on a Real Server with the agent installed will return the current idle stats as an integer value in the range 0 - 100. The figure returned can be related to CPU utilization, RAM usage or a combination of both. This can be configured using the XML configuration file located in the agents installation folder (by default C:\ProgramData\LoadBalancer.org\LoadBalancer).

The load balancer typically expects a 0-99 integer response from the agent which by default relates to the current CPU idle state, e.g. a response of 92 would imply that the Real Servers CPU is 92% idle. The load balancer will then use the formula (92/100*requested_weight) to find the new optimized weight.

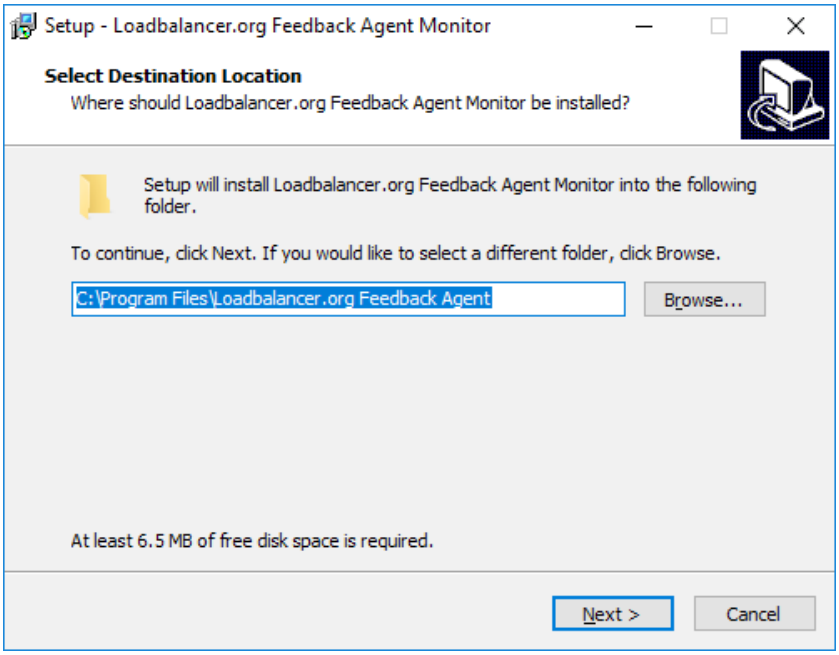
Note

The 'Requested Weight' is the weight set in the WebUI for each Real Server.

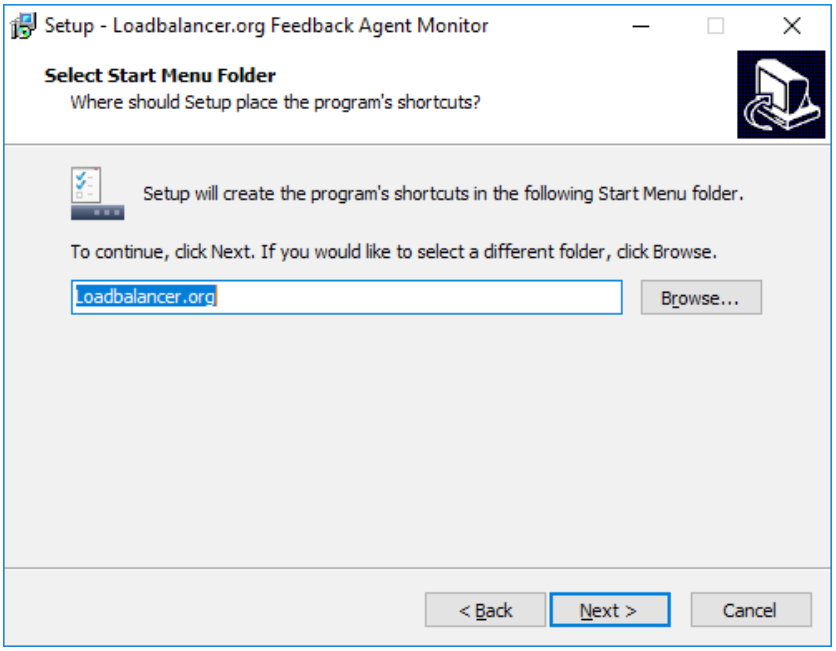
For more information about the feedback agent please refer to this blog.

Windows Agent

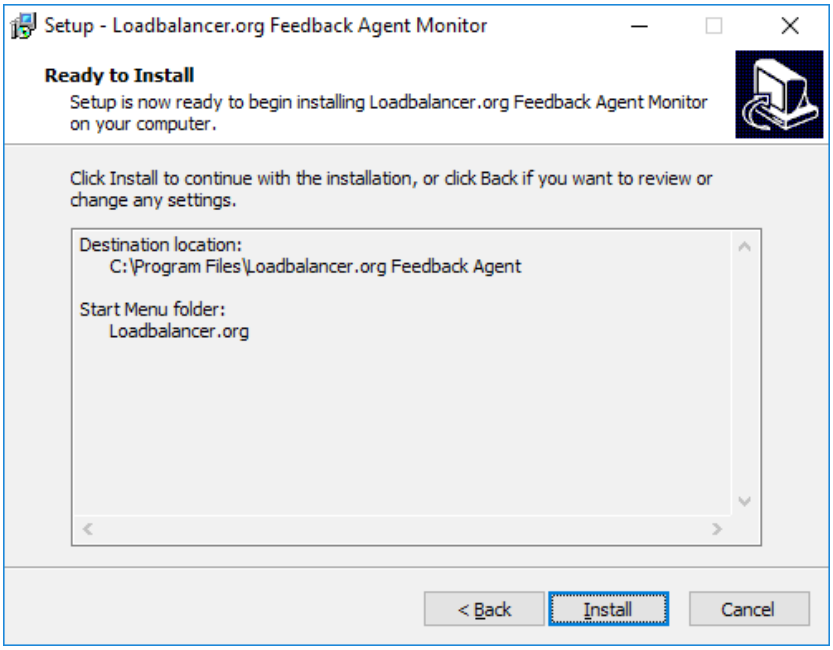
The latest Windows feedback agent can be downloaded from here. To install the agent, run **loadbalanceragent.msi** on each Real Server:



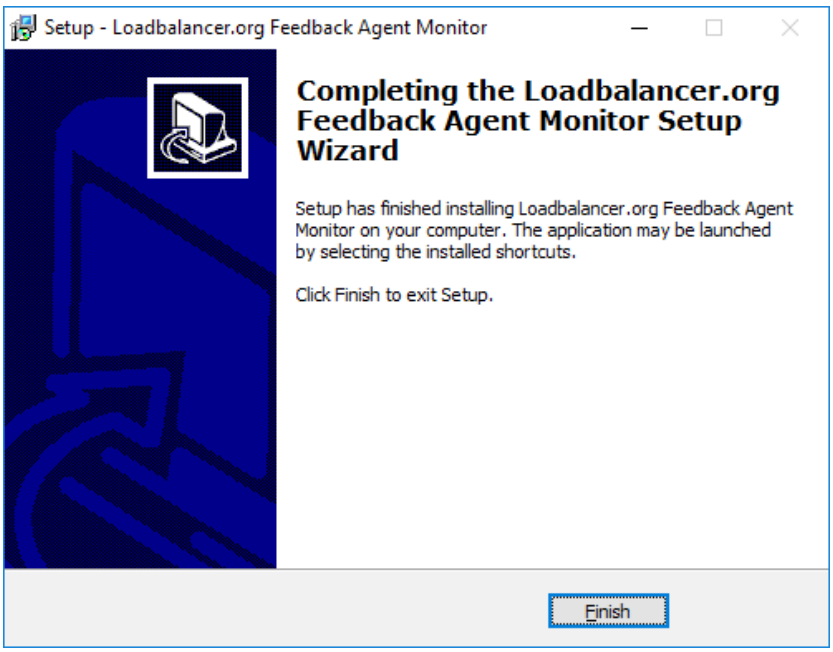
Leave the default location or change according to your requirements, click **Next**.



Leave the default location or change according to your requirements, click **Next**.



Click **Install** to start the installation process.

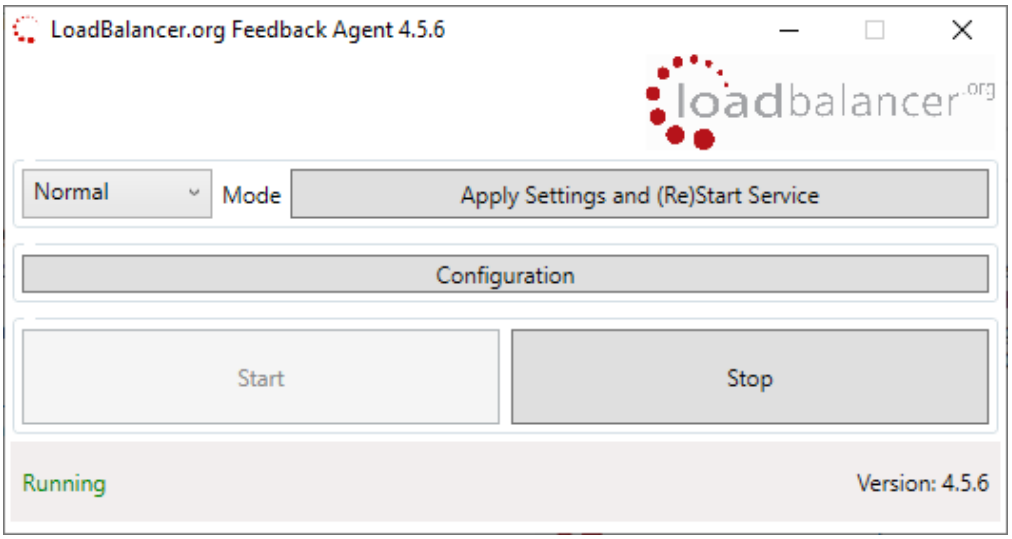


Click **Finish**.

Note | The agent should be installed on all Real Servers in the cluster.

Starting the Agent

Once the installation has completed, you'll need to start the service on the Real Servers. The service is controlled by the Feedback Agent monitor & control program that is also installed along with the Agent. This can be accessed on the Windows server from: *Start> Loadbalancer.org > Loadbalancer.org Feedback Agent.* It's also possible to start the service using the services snap-in - the service is called **LBCPUMon**.



- To start the service, click the **Start** button
- To stop the service, click the **Stop** button

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:

```
apt-get install xinetd (if not already installed)
```

insert this 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:

```
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 all Real Servers, not the load balancer.

Custom HTTP Agent

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 applications requirements.

Configuring VIPs To Use The Agent

As mentioned, both layer 4 and layer 7 VIPs can be configured to use the feedback agent. To Configure Virtual Services to use Agent/HTTP Feedback follow the steps below:

1. Using the WUI, navigate to:
   - *Cluster Configuration > Layer 4 - Virtual Services* ***Or***
   - *Cluster Configuration > Layer 7 - Virtual Services*
2. Click **Modify** next to the relevant Virtual Service

3. Change the Feedback Method to either **Agent** or **HTTP** for layer 4 VIPs
4. Change the Feedback Method to **Agent** for layer 7 VIPs
5. Click **Update**
6. Reload/restart services as prompted

Global Server Load Balancing (GSLB)

GSLB functionality has been added to the appliance using the Open Source Polaris GSLB. When used in conjunction with the usual failover and high availability features of the appliance, GSLB extends the options available to create a highly available load balanced environment.

Key Concepts

When configured, 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 (please see note below) 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 2 site setup with an HA pair of load balancers in each site, once GSLB and DNS delegation are correctly configured, the 4 load balancers act as intelligent name servers for the sub domains in question.

Note

When topology is configured, it is the IP address of the client's local DNS server that is used to determine relative location. Therefore, if multiple client sites use the same DNS server, it's not be possible for GSLB to distinguish between locations. If this is required, a local DNS server for each client site must be configured.

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

Note

For additional background and configuration information about GSLB, please refer to our blog.

GSLB Configuration

Prior to v8.5, GSLB was configured by editing the Polaris and Topology configuration files via the WebUI. For V8.5 and later, a series of configuration screens enables GSLB to be configured without having to directly modify the underlying configuration files. This greatly simplifies the process of configuring GSLB. Configuration is performed using the WebUI menu option: *Cluster Configuration > GSLB Configuration*.

As illustrated above, 4 tabs are used to configure GSLB:

Global Names
Global Names are used to define the FQDN's that GSLB responds to.

Members
Members, also known as 'endpoints' are returned to clients in DNS responses.

Pools
A Pool links together a Global Name and the relevant members and also defines the heath checks, timeouts and other settings that should be used.

Topologies
Topologies define how network subnets map to sites. In a multi-site deployment, this is used to define which site clients should connect to under normal conditions.

The following table describes the options in each tab.

Tab	Setting	Description
Global Names	Name	Name can be a combination of '0-9', 'a-z', 'A-Z', '-' (dash), '_' (underscore) or a '.' (dot).
	Hostname	A valid RFC 1123 hostname, for example www.example.com
	TTL	TTL is how long to cache the (hostname) DNS response in seconds. For example 3600 would be equal to 1 hour, minimum value is "1" (1s).
Members	Name	Name can be a combination of '0-9', 'a-z', 'A-Z', '-' (dash), '_' (underscore) or a '.' (dot).
	IP	A valid IPv4 address for example 10.0.1.1
	Monitor IP	A valid IPv4 address for example 10.0.1.1
	Weight	Weight of the server, min: 0 (server is disabled), max: 10
Pools	Name	A Name can be a combination of '0-9', 'a-z', 'A-Z', '-' (dash), '_' (underscore) or a '.' (dot).
	Monitor	The type of health check to use. The options are: • **http** - perform HTTP(S) GET, succeeds if response HTTP status is 200(or one of expected_codes is specified). • **tcp** - Perform a TCP connect. (Optionally: send text, read response, match a reg exp pattern). • **forced** - Forces a member to be either UP or DOWN effectively disabling the health checking. • **external** - Will run the script selected in the *Monitor Script* drop-down. The check will receive the IP address of the member the port 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. For more information please refer to External Health Check Scripts (GSLB). • **external dynamic weight** - 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 healthcheck failure.
	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, default is "none".
	Monitor URL Path	A url path to request, appended after the member's IP address, default is "/".

Tab	Setting	Description
	Monitor Port	An integer between 1 and 65535, if value is not provided, port 80 will be used with use_ssl set to false, port 443 will be used with use_ssl set to true.
	Monitor Expected Codes	An array of HTTP codes to match in a response for example "200", "301". Input range is between 100 and 599
	LB Method	The load balancing method to use. The options are: • **wrr** - weighted round-robin • **twrr** - topology weighted round-robin • **fogroup** - failover group, when this method is assigned, IP address of the first configured member that is healthy is handed out continuously, unless the member becomes unhealthy, then, the next healthy member IP is handed out etc.
	Global Names	A Pool can be associated with one or more global names, a pool requires at least one global name. Press "CTRL" and "click" to select multiple "globalnames".
	Members	A pool must have at least one endpoint member. Drag and drop the endpoints. NOTE: If using fogroups the order is important.
Pools (Advanced)	Monitor Interval	In seconds, min: 1, max: 3600
	Monitor Timeout	In milliseconds, min: 100 (0.1ms), max: 10000 (10 seconds)
	Monitor Retries	Retry min: 0, max: 5, default is '0' no retries.
	Fallback	Resolution behavior when all members of the pool are DOWN. The options are: • **any** - default, perform distribution among all the configured members with non-0 weight(ignore health status) • **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.
	Max Addresses Returned	Maximum number of A records to return in response, large responses will go over TCP min: "1", max: "1024", default: "1".

External Health Check Scripts (GSLB)

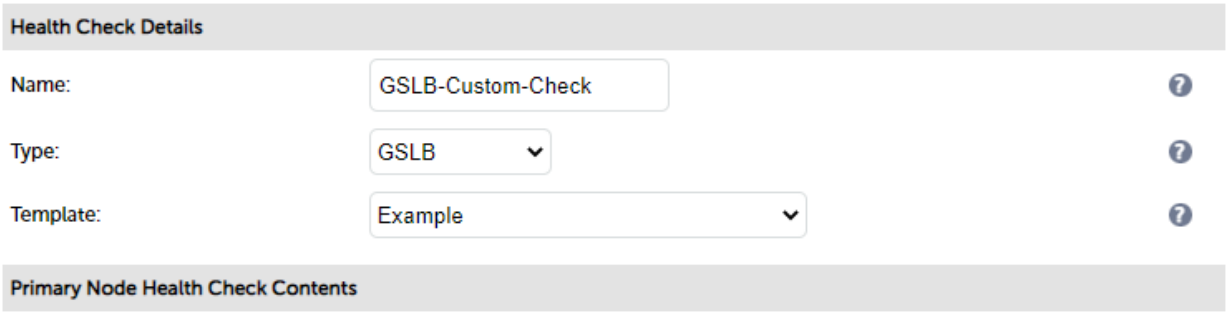
From v8.6, custom GSLB health checks can be created and modified directly from within the WebUI. Previous versions required scripts to be created using an editor and then saved to a specific location on the appliance - this enabled the script to be selectable when configuring external health checks.

Adding Health Check Scripts

New scripts can be created either by using the script templates or by uploading files from an external source.

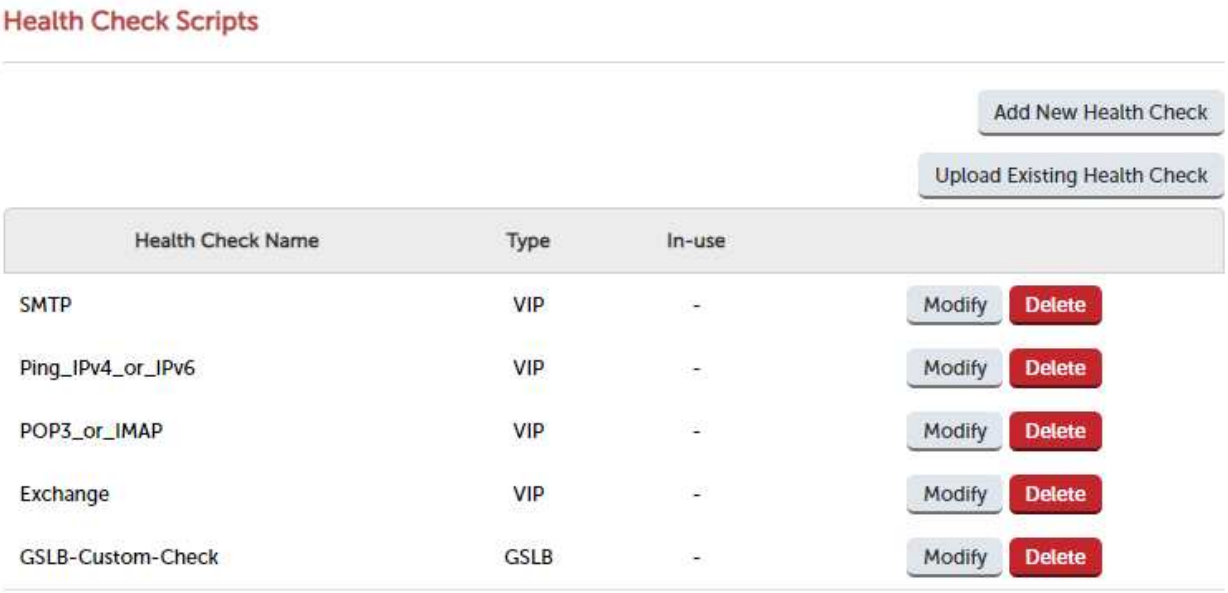
To Create a new Script From Template:

1. Using the WebUI, navigate to *Cluster Configuration* > *Health Check Scripts* and click **Add New Health Check**.

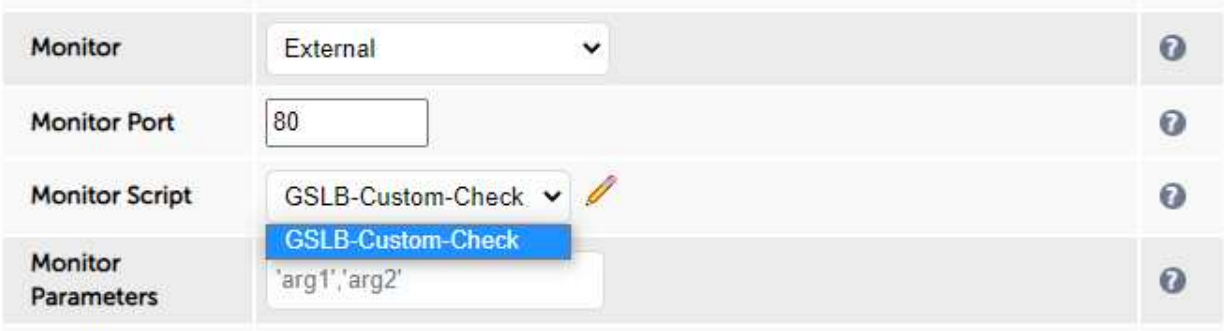


2. Specify an appropriate *Name* for the health check, e.g. **GSLB-Custom-Check**.
3. Set *Type* to **GSLB**.
4. Using the *Template* dropdown select an appropriate template from the **GSLB** section of the list, e.g. **Example**.
5. Modify the script to suit your requirements.
6. Click **Update**.

Once the health check has been added, it will appear in the Health Check Scripts list as shown below:



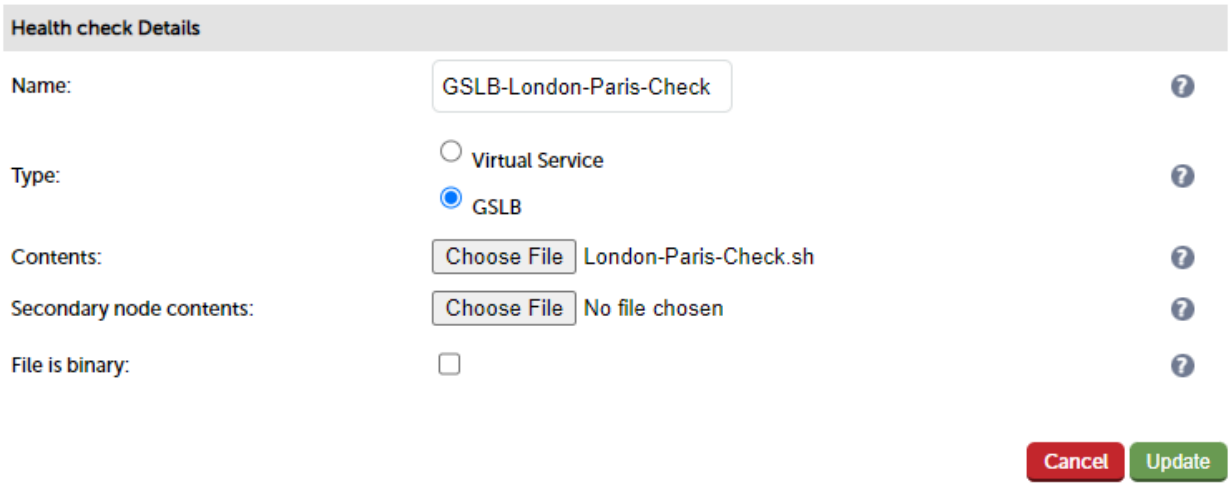
The new script will also appear in the *Monitor Script* dropdown when *Monitor* for a Pool is set to **External**:



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**.



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.

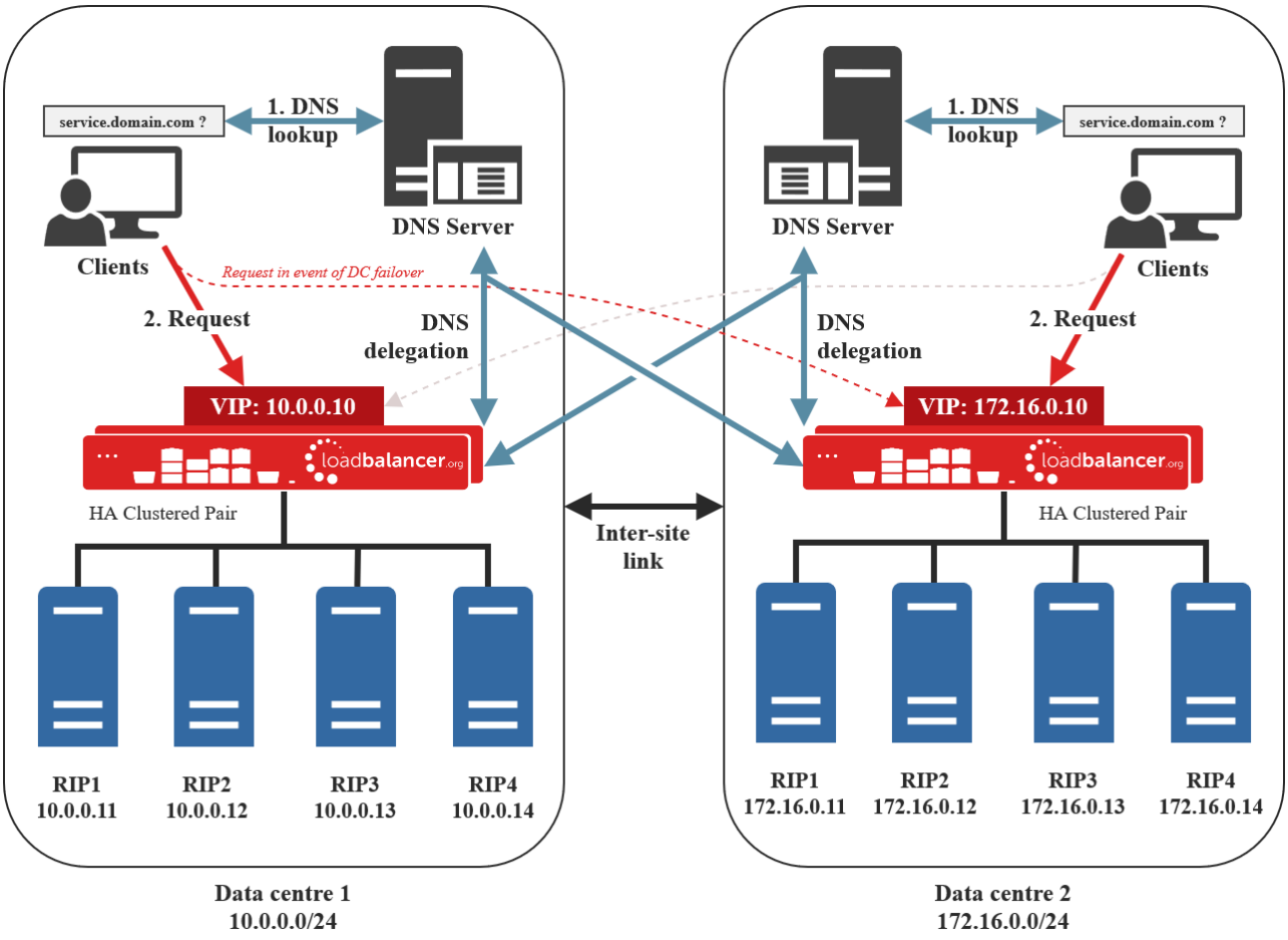
GSLB Multi-site Example

This example demonstrates the steps required to configure GSLB for a 2 site deployment. It's possible for this to be extended to encompass up to 16 sites.

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

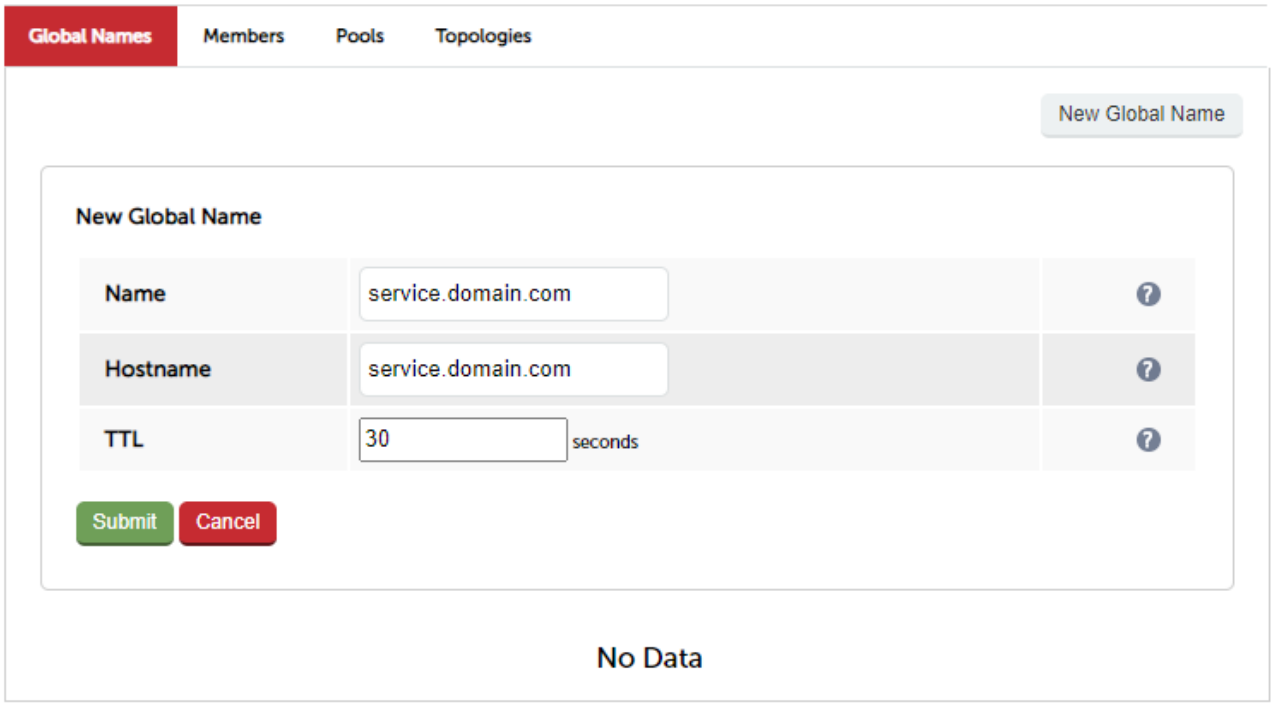
In the event that the load balanced cluster and/or load balancers at one site should completely fail then local clients will be directed to the load balanced cluster at the other site and the service will continue to be available. This style of multi-site failover is possible because the load balancer's 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 across all 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.

Step 1 – Configure the Global Name

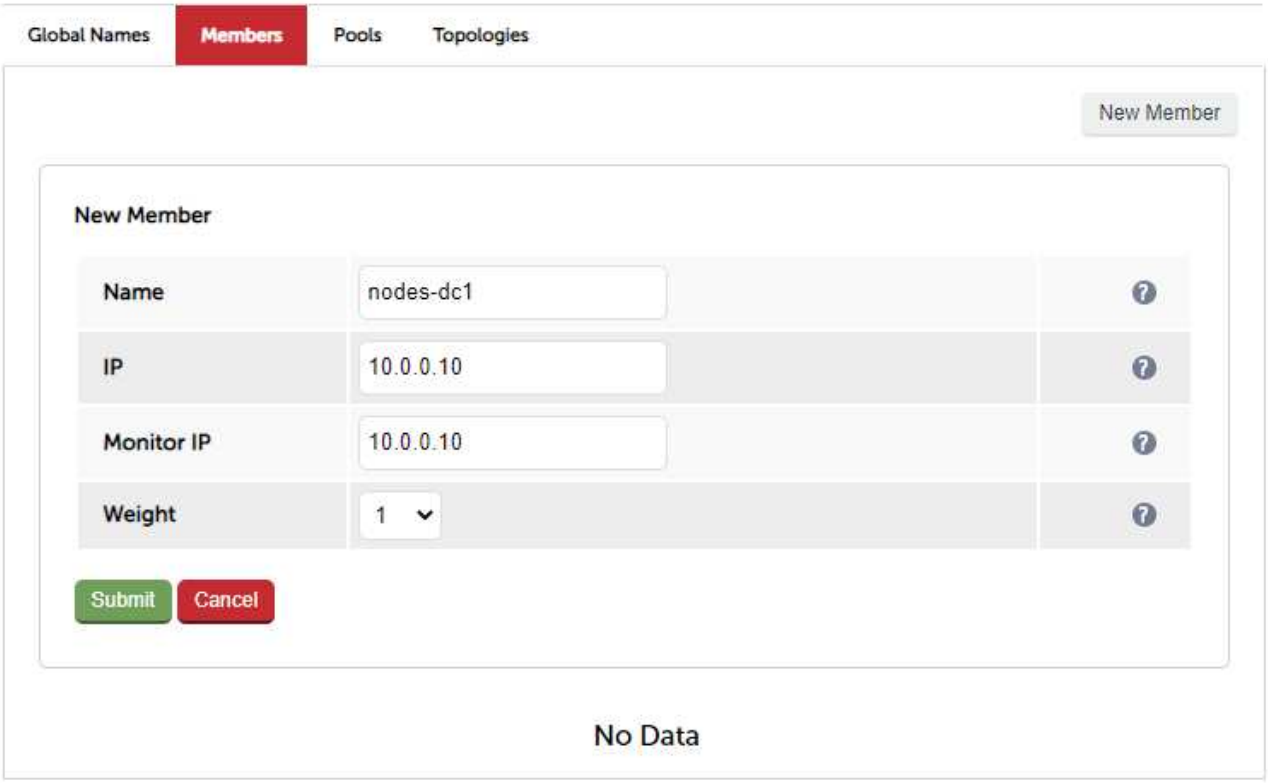
1. Using the WebUI on the Primary appliance for data center 1, navigate to: *Cluster Configuration* > *GSLB Configuration*.
2. Select the *Global Names* tab.



3. Define the required *Name* and *Hostname*, in this example both are set to **service.domain.com**.
4. Set the TTL as required, the default of **30** seconds is appropriate in many cases.
5. Click **Submit**.

Step 2 – Configure the Members

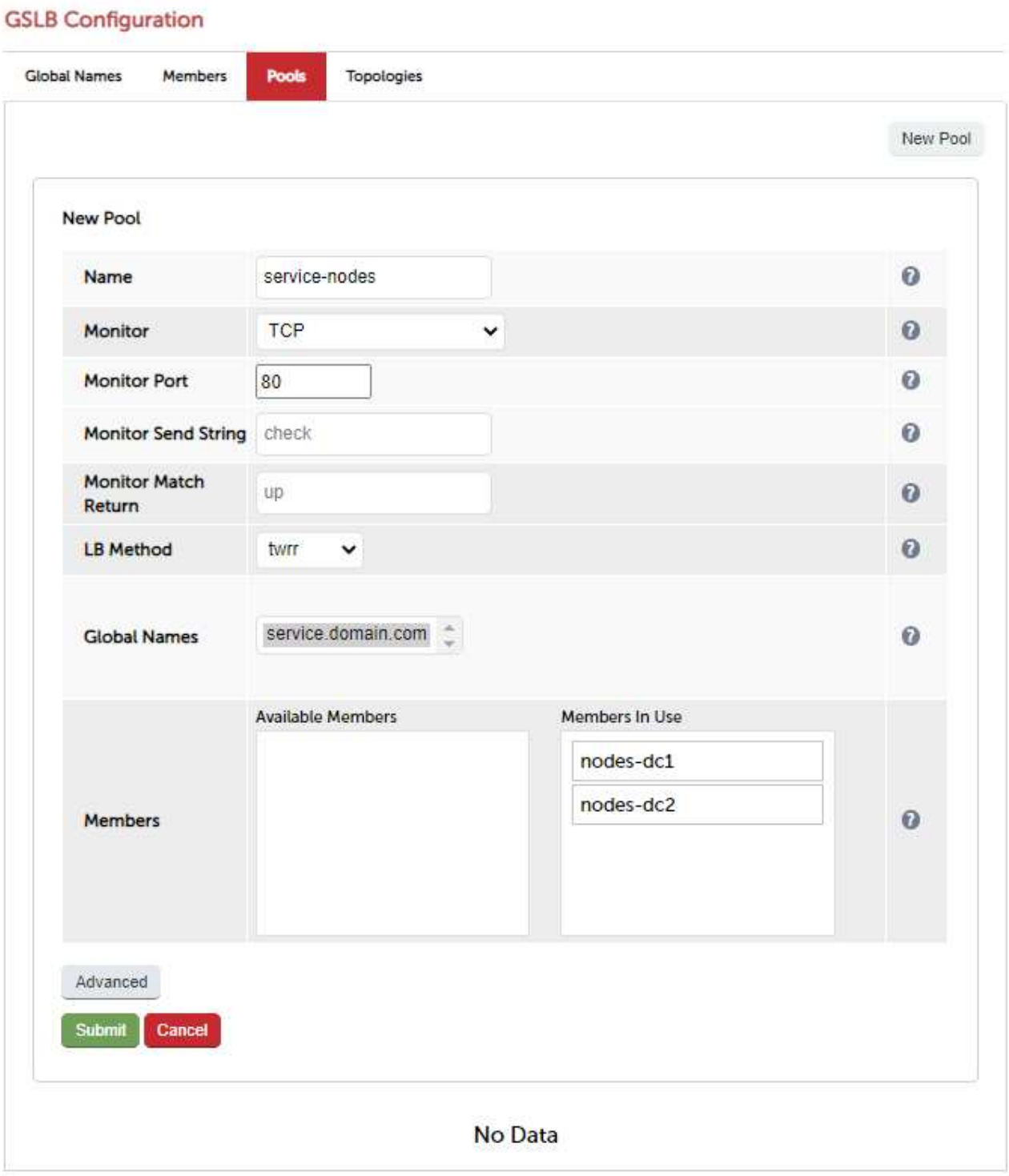
1. Select the *Members* Tab.
2. Click the **New Member** button.



3. In this example, 2 members must be created - one for the VIP in Data Center 1 and the second for the VIP in Data Center 2. Enter a *name* for the first member, e.g. **nodes-dc1** and specify the IP and Monitor IP, in this example **10.0.0.10**.
4. Leave the *weight* set to **1**.
5. Click **submit**.
6. To create the second member, click the **New Member** button again and define the second member (**nodes-dc2**) in the same way.

Step 3 – Configure the Pool

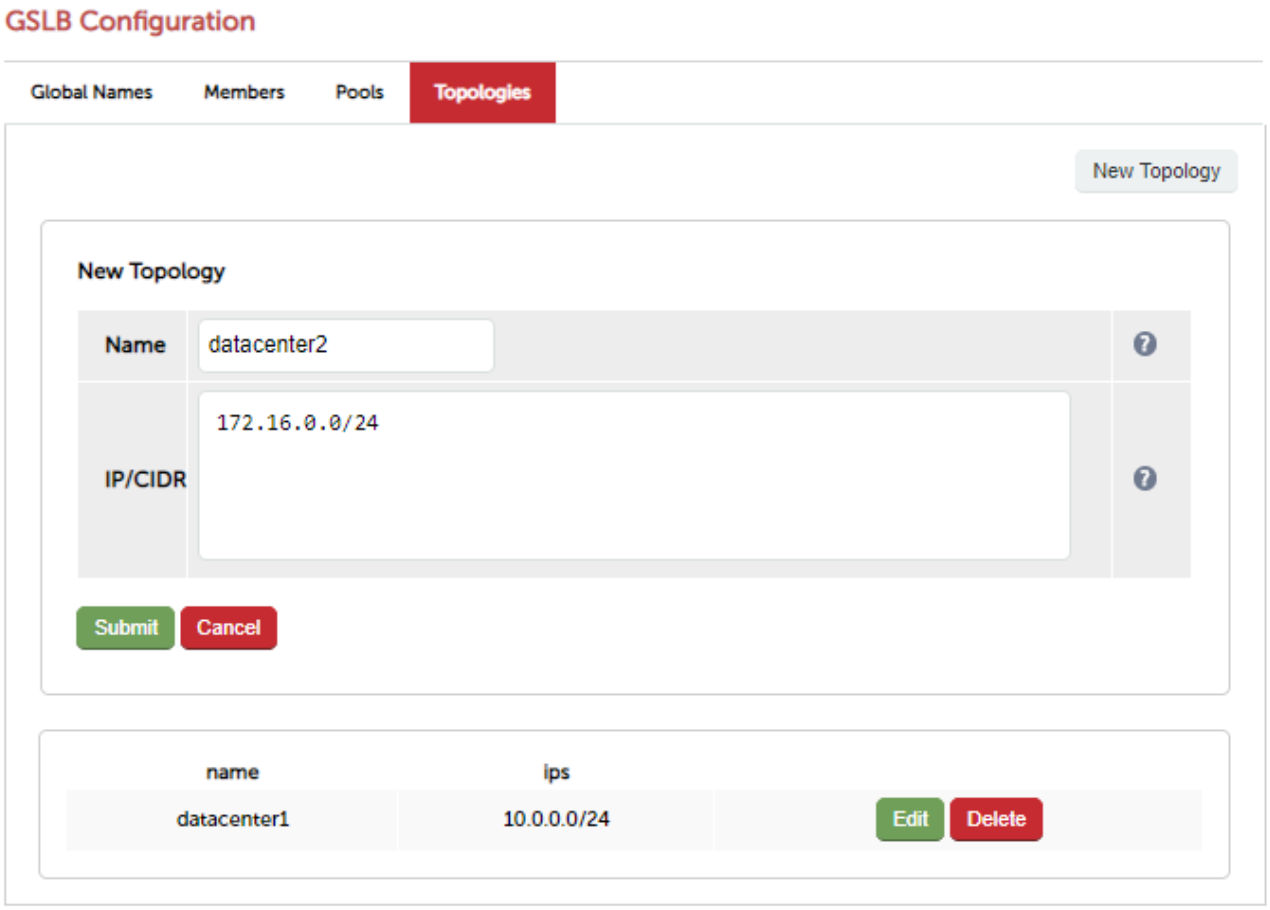
1. Select the *Pools* Tab.



2. Enter a suitable name for the Pool, in this example **service-nodes**.
3. Set the *monitor* to **TCP** – this will perform a basic TCP port connect to verify each member.
4. Set the *monitor port* to the required value, in this example **80**.
5. Set the *LB Method* to **twrr** (Topology Weighted Round Robin).
6. Select the required *Global Name*.
7. Drag the required Members from the Available Members list to the Members in Use list, in this example **nodes-dc1** and **nodes-dc2**.
8. Click **Submit**.

Step 4 – Configure the Topology

1. Select the *Topologies* Tab.



2. 2 Topologies must be created, one for Data Center 1 and one for Data Center 2. Enter a *Name* for the first Topology, e.g. **datacenter1** and specify the *IP /CIDR*, in this example **10.0.0.10/24**.
3. Click **Submit**.
4. To create the second Topology, click the **New Topology** button and define the second topology (**datacenter2**) in the same way.

Note	Since this example has 2 data canters with an HA pair in each, you now also need to configure GSLB in the same way for the second pair using the WebUI on the Primary appliance in data center 2 ensuring that the configuration is identical.

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

2 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.

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.

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 various appliance features to be configured and controlled. A JSON based Application Programming Interface (API) has also been added that enables CLI commands to be called from a Web Service.

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)

The CLI is called using the lbcli command:

```
Usage: lbcli --action <action> --(option 1) (value) --(option 2) (value)....
```

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**.

Appliance Status Actions

Get node status:

```
lbcli --action nodestatus
```

System Overview Actions

Drain a Server:

```
lbcli --action drain --vip <VIP Name> --rip <RIP Name> --services all
```

Halt a Server:

```
lbcli --action halt --vip <VIP Name> --rip <RIP Name> --services all
```

Online a Server:

```
lbcli --action online --vip <VIP Name> --rip <RIP Name> --services all
```

--services all is optional to perform the same --action accross all VIPs.

This will only work for the layer the defined VIP is in, it will not online halt or drain RIPs from another layer.

e.g. If the first VIP is a Layer7 VIP then all the real servers in Layer7 with the same --rip label will have the same action applied.

VIP Actions

Add a VIP:

Layer 4

```
lbcli --action add-vip --layer 4 --vip <VIP Name>
--ip <VIP IP Address>
--ports <ports>
--forwarding <gate|masq|ipip|snat>
--protocol <tcp|udp|tcpudp|ops|fwm>
--slave_ip <1.2.3.4> # Slave IP is for Azure only
```

Layer 7

```
lbcli --action add-vip --layer 7 --vip <VIP Name>
--ip <VIP IP Address>
--ports <ports>
--mode <http|tcp>
--fallback_ip <IP Address>
--fallback_port <port>
--service_type <waf_frontend>
--slave_ip <1.2.3.4>
--encrypt_all_backends <on:off>
```

Delete a VIP:

```
lbcli --action delete-vip --vip <VIP Name>
```

Edit a VIP:

Warning	When using the edit VIP be aware you can break your configuration. Take care to use the right combination of options. All possible options are shown for Layer4 and Layer7.

```
lbcli --action edit-vip --vip <VIP_NAME_TO_EDIT>
```

Layer 4

```
--ip <IP Address of the VIP>
--ports (Ports can be 80 80:81 or 800-900 or 80:90-100:3443 as a mix of port:separated:values
and also port-ranges values)
--protocol <tcp:udp:ops:fwm> # We do not support manual firewall marks where IP = FWM Number
as you need to manually add the firewall rules
--forwarding <gate:masq:ipip> Gate = L4 DR masq=L4 NAT ipip=TUN Mode
--granularity <255.255.255.255> This is the subnet or single ip range for persistence
--fallback_ip <127.0.0.1> This is the fallback server IP Address, It may be an external IP
Address
--fallback_port <9081> This is the fallback server port, it may be the port of an external web
server.
--fallback_local <on:off> MASQ Fallback. Allows fallback server port to be different to that
of the real server.
--persistent <on:off> Are we a persistent Layer4 VIP , this is simply on or off
--persist_time <300> The persistent time in seconds by default is 300
--scheduler <wlc:wrr:dh> wlc=Weighted Least Connection, wrr=Weighted Round Robin, dh
=Destination Hash
--feedback <agent:http:none> agent=Feedback Agent, http=HTTP, none=No Feedback
--email <recpt@email.com> Your email address to receive email alerts
--email_from <sender@email.com> Sending email address of email alerts
--check_service
<http:https:http_proxy:imap:imaps:pop:pops:ldap:smtp:nntp:dns:mysql:sip:simpletcp:radius:none>
If check type = Negotiate then Layer4 knows about various service
--check_vhost <host header> When using a Negotiate check we can enable a host header to check
a known site status used for HTTP,HTTPS
--check_database <db> Database to check if check_service=mysql
--check_login <username> used when check_service is MySQL,FTP,IMAP,IMAPS,POP,POPS,LDAP,SIP
--check_password <password> This is the password used with the check_login when required,
FTP,IMAP,IMAPS,POP,POPS,LDAP,MYSQL,SIP
--check_type <negotiate|connect|ping|external|off|on|5|10> This is the check type, Negotiate,
Connect to port, External script, no checks, always off, No checks, always on, 5 Connects, 1
Negotiate, 10 Connects, 1 Negotiate
--check_port <80> Port to check when using Negotiate check
--check_request <check.txt> used for check_service= http, https
--check_response <OK> Response expected to the check_request
--check_secret <secret> This is used only if check_service = RADIUS
--check_command <external_script.sh> This is used when check_type=external
--autoscale_group <YOUR AUTO SCALE GROUP NAME> if in AWS the name of the auto scale group you
have defined.
```

Layer 7

```
--ip <IP Address of VIP>
--ports (Ports can be 80 80:81 or 800-900 or 80:90-100:3443 as a mix of port:separated:values
and also port-ranges values)
--mode <http:tcp> Mode of the Layer7 VIP it is either http or tcp, tcp is an alias of
other_tcp and either can be specified
--persistence <http:appsession:sslsesid:rdp-session:rdp-
cookie:ip:http_ip:xff:none:fallback_persist>
```

Note	Persistence modes available in tcp mode: sslsesid, appsession, rdp-session, ip. Persistence modes available in http mode: http, appsession, rdp-cookie, ip, http_ip, xff. When using '--persistence fallback_persist' you MUST set --fallback_ip and --fallback_port otherwise it will fail.

```
--cookiename <SERVERID> only available when persistence is http,http_ip
```

```
--fallback_ip <127.0.0.1> Fallback Server IP Address, this is either the internal NGINX
fallback or external or VIP of fallback server
--fallback_port <9081> Fallback Port, 9081 by default of that of the external fallback server
ports
--persist_time <30> Persistence timeout available when persistence=appsession,sslsesid,rdp-
cookie,ip,http_ip,xff
--persist_table_size <10240> Persistence table size available when persistence
=appsession,sslsesid,rdp-cookie,ip,http_ip,xff
--maxconn <40000> max conns allowed to the VIP
--scheduler <roundrobin:leastconn> Weighted Round Robin or Weighted Least Connections
--check_port <Port of Service> Check port is available when check is
negotiate_http,negotiate_https,connect,mysql
--check_request <check.txt> name of file to request
--check_receive <OK> response expected from check request
--check_host <VHOST> Check host header for checking a virtual host with host header
--check_username <mysql> Healthcheck username, only available with check type=mysql
--appsession_cookie <JSESSIONID:PHPSESSIONID:ETC> The application session ID provided by your
real server.
--forward_for <on:off> Insert X-Forward-For only available in http mode.
--http_pipeline <http_keep_alive|http_close|http_server_close|http_force_close> This is only
available in mode=http
--http_pretend_keepalive <on:off> Work around broken connection: close  This is only available
in mode=http
--stunneltproxy <on:off> Only select on if behind an STunnel ssl termination and where STunnel
proxy is also enabled on the SSL Termination
--feedback_method <agent:none> The feedback method is either the feedback agent or none. This
is available in mode http or tcp
--fallback_persist <on:off> Is the fallback server persistent on or off
--feedback_port <3333> Port used for the feedback agent by befault is 3333 only when method
=agent
--check_type
<negotiate_http:negotiate_http_head:negotiate_https:negotiate_https_head:connect:external:mysq
l:none> Type of health check to use negotiate_https or negotiate_httpd_head are only available
when backend is encrypted
--external_check_script <scriptname.sh> This is the filename of external check scripts in
/var/lib/loadbalancer.org/check/ available when check_type=external
--tcp_keep_alive
--force_to_https <on:off> Force connection to https, if used then no other options need be
configured and no real servers need be present in the VIP. take care when using STunnel_proxy
=on
--timeout <on:off> Enable or disable client / real server timeout
--timeout_client <12h> Client Timeout by default 12 hours
--timeout_server <12h> Real Server Timeout by default 12 hours
--redirect_code <301:302:303:307:308> Only used if force_to_https=on 301 (Moved Permanently),
302 (Found), 303 (See Other), 307 (Temporary Redirect), 308 (Permanent Redirect)
--no_write <on:off> This is used to enable manual configuration of the VIP. Not suggested for
full lbcli use as you can not edit the manual configuration unless you upload it manually
--waf_label <WAF_VIP_NAME> When creating a WAF the WAF Service will add this to the VIP, Care
needs to be taken when changing this as the WAF also needs updating
--clear_stick_drain <on:off> Do you want to clear the stick table on drain of the RIP in the
VIP
--compression <on:off> Do we enable compression on the VIP, only available in mode=http
--autoscale_group <YOUR AUTOSCALE GROUP NAME>  if in AWS the name of the autoscale group you
have defined
--cookie_maxidle <30m> Cookie Max Idle Duration
--cookie_maxlife <12h> Cookie Max Life Duration
--source_address <192.168.2.21> IP Address used for health check source IP
--backend_encryption <on:off> Only available on mode=http. Do we want to re-encrypt to the
real server?
--enable_hsts <on:off> Only available in mode=http
--hsts_month <6> Months the HSTS is valid 3-24 months, Only available in mode=http
--xff_ip_pos <-1> Move the XFF header back one in the list to show client IP in correct place.
This is only available when persistence=xff
--invalid_http <on:off> Accept invalid http requests. this is only available in mode=http
--send_proxy <none:v1:v2:v2_ssl:v2_ssn_cn>  Send Proxy Protocol None, Send Proxy V1, Send
Proxy V2, Send Proxy V2 SSL, Send Proxy V2 SSL CN
--as_port <1234> Autoscale Port on the real servers you have defined in AWS
--http_request <on:off> Default is on to enable Slowlaris protection. You would usually not
need to disable this unless the headers are delayed more than 5 seconds
--stunnel_source <1.2.3.4> Source IP of STunnel VIP
--proxy_bind <name of Layer7 VIP>  Name of the Layer7 VIP to bind to.
--slave_ip <1.2.3.4> #Azure Only
--tunneltimeout Value in seconds for WebSockets
--redispatch <on:off> turn redispatch on or off
--fallback_encrypt <on:off> Encrypt connection to the fallback server if it is a TLS
Connection
--http_reuse_connection <on:off> It is possible to reuse idle connections to serve requests
from the same session which can be beneficial in terms of performance. It is 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.
--tproxy <on:off> Turn tproxy on and off on a VIP level.
```

Clone VIP

```
lbcli --action clone-vip
--vip <VIP to Clone>
--clone <New Cloned VIP>
--ip <new IP Address> |or| --ports <new ports>
```

Optional combination of --ip --ports to change IP address or pots of the cloned VIP. If you use any option in edit-vip the cloned VIP will also be updated to reflect the syntax supplied.

```
example: lbcli --action clone-vip --vip source --vip dest --ip 192.168.100.120 --ports 443
--persistence ip --mode tcp
```

This will set the persistence to IP and mode to tcp, it may have been mode http in the source VIP.

RIP Actions

Add a RIP:

Layer 4

```
lbcli --action add-rip
--vip <VIP Name>
--rip <RIP Name>
--ip <RIP IP Address>
--weight <Weight value>
--port <Port Value>
--minconns <minconns>
--maxconns <maxconns>
```

Layer 7

```
lbcli --action add-rip
--vip <VIP Name>
--rip <RIP Name>
--ip <RIP IP Address>
--weight <Weight value>
--port <Port value>
--minconns <minconns>
--maxconns <maxconns>
--encrypted <on|off>
```

Delete a RIP:

```
lbcli --action delete-rip --vip <VIP Name> --rip <RIP Name>
```

Edit a RIP:

```
lbcli --action edit-rip
--vip <VIP Name>
--rip <RIP Name>
--ip
--port
--weight
--minconns
--maxconns
--encrypted # layer 7 VIPs
```

WAF Actions

Add a WAF:

```
lbcli --action add-waf --vip <VIP Name> --waf <WAF Name>
```

Edit a WAF:

```
lbcli --action edit-waf --waf <WAF Name>
--in_anom_score <1:100>
--out_anom_score <1:100>
--req_data <on:off>
--resp_data <on:off>
--audit <on:off>
--proxytimeout
--dlogin <on:off>
--dlogin_mode <static:openid_google>
--dlogin_location </:/dir:/file.html>
--dlogin_static_username <username>
--dlogin_static_password <password>
--dlogin_google_clientid <Google API Client ID>
--dlogin_google_clientsecret <secret>
--dlogin_google_redirect_uri <redirect uri>
--dlogin_google_passphrase <passphrase>
--dlogin_google_allowed_domain <example.com email domain>
--rule_engine <on:off>
--disable_waf <on|off>
--cacheaccel <on|off>
--cache_nocache_files <file or regex>
--cache_force_cache <on|off>
--cache_object_size
```

Delete a WAF:

```
lbcli --action delete-waf --vip <VIP Name> --waf <WAF Name>
```

Floating IP Actions

Add a FIP:

```
lbcli --action add-floating-ip --ip <IP Address>
```

Delete a FIP:

```
lbcli --action delete-floating-ip --ip <IP Address>
```

Fix FIP:

```
lbcli --action fix-floating-ip --ip <IP Address>
```

fix-floating-ip removes and re-adds the floating IP, care is needed if the IP is up on the wrong interface and it has other base IP addresses within the same subnet. we will get confused. First ip ad del x.x.x.x/cidr dev ethx to remove all IPs on the wrong interface then run lbcli --action fix-floating-ip --ip x.x.x.x.

Service Actions - Restart

Restart Ldirectord:

```
lbcli --action restart-ldirectord
```

Restart HAProxy:

```
lbcli --action restart-haproxy
```

Restart Heartbeat:

```
lbcli --action restart-heartbeat
```

Restart Pound:

```
lbcli --action restart-pound
```

Restart STunnel:

```
lbcli --action restart-stunnel
```

Restart Collectd:

```
lbcli --action restart-collectd
```

Restart Firewall:

```
lbcli --action restart-firewall
```

Restart Syslog;

```
lbcli --action restart-syslog
```

Restart SNMPD:

```
lbcli --action restart-snmp
```

Restart WAF:

```
lbcli --action restart-waf
```

Restart AWS Autoscaling:

```
lbcli --action restart-autoscaling
```

Restart AWS Availability Zone HA:

```
lbcli --action restart-azha
```

Service Actions - Reload

Reload Apache WUI:

```
lbcli --action reload-apache
```

Reload Ldirectord:

```
lbcli --action reload-ldirectord
```

Reload HAProxy:

```
lbcli --action reload-haproxy
```

Reload WAF:

```
lbcli --action reload-waf
```

Reload Syslog:

```
lbcli --action reload-syslog
```

Reload STunnel:

```
lbcli --action reload-stunnel
```

Reload Heartbeat:

```
lbcli --action reload-heartbeat
```

HAProxy Stick Table Actions

Clear:

```
lbcli --action haproxy-clear-stick
```

SSL Related Actions

List Certificates:

```
lbcli --action termination --type certificate --function list
```

Create CSR:

```
lbcli --action termination --type certificate --function csr
--csrname <CSRNAME>
--city <CITY>
--province <COUNTY>
--country <ISO COUNTRY CODE : GB for UK>
--organisation <ORG>
--unit <UNIT>
--domain <example.com>
--email <ssl@example.com>
--csrsize <2048:4096>
--signalgorithm sha256
--days
```

Upload SSL PEM/PFX: Please refer to this blog: https://www.loadbalancer.org/blog/how-do-i-automate-load-balancer-deployments/

Add SSL Termination (STunnel):

```
lbcli --action termination --type stunnel --function add  --vip <VIPNAME>
--ip <IP ADDRESS>
--port <PORT>
--backend_ip <BACKEND IP>
--backend_port <BACKEND PORT>
--sslcert <SSLCERTNAME>
--slave_ip <Azure Only>
--disabletlsv1_1  <on:off>
--disabletlsv1_2 <on:off>
--disabletlsv1_3 <on:off>
--sslmode <high|fips|compatable|custom>
--haproxy_ssl_link (This is a combination of VIP_Name^VIP^PORT or custom>
```

Edit Termination:

```
lbcli --action termination --type stunnel --function edit  --vip <VIPNAME>
--ip <IP ADDRESS>
--port <PORT>
--backend_ip <BACKEND IP>
--backend_port <BACKEND PORT>
--sslcert <SSLCERTNAME>
--sslmode <high|fips|compatable|custom>
--haproxy_ssl_link (This is a combination of VIP_Name^VIP^PORT or custom)
--ciphers \# this and the syntax below are optional
--disablesslv2 <on:off>
--disablesslv3 <on:off>
--disabletlsv1 <on:off>
--stunneldnsdelay <on:off>
--stunnelproxy <on:off>
--servercipherorder <on:off>
--emptyfragments  <on:off>
--stunnelrenegotiation <on:off>
--stunneltimetoclose 0
--proxy_bind
--slave_ip
--disabletlsv1_1 <on:off>
--disabletlsv1_2 <on:off>
--disabletlsv1_3 <on:off>
--sslcert server  : "server" # is the inbuilt default SSL Certificate
```

Delete Termination:

```
lbcli --action termination --type stunnel --function delete --vip <VIPNAME>
```

SSL SNI Actions

Add SNI Rules:

```
lbcli --action termination --type stunnel --function edit --vip <VIP> --sni add
--sni_name <SNINAME>
--sni_rule <example.com>
--sni_cert <SSLCERTNAME>
--sni_backend_proxyprotocol <on:off>
--sni_backend_service <L7VPIName>
--sni_backend_ip <SNI_BACKEND_IP>
--sni_backend_port <BACKEND_PORT>
```

Edit SNI Rules:

```
lbcli --action termination --type stunnel --function edit --vip <VIP> --sni edit
--sni_name <Existing SNINAME>
--sni_rule <example.com>
--sni_cert <SSLCERTNAME>
--sni_backend_proxyprotocol <on:off>
--sni_backend_service <L7VPIName>
# or use below to replace sni_backend_service name and define ip and port
--sni_backend_ip <SNI_BACKEND_IP> --sni_backend_port <BACKEND_PORT>
```

Delete SNI Rules:

```
lbcli --action termination --type stunnel --function edit --vip <VIP> --sni delete
--sni_name <SNINAME>
--sni_rule <example.com>
```

Layer 7 ACL Actions

List ACL Rules:

```
lbcli --action acl --function list --vip <VIPNAME>
```

Add ACL Rules:

```
lbcli --action acl --function add --vip <L7VIPNAME>
--pathtype <path_beg|path_end|hdr_host|hdr_beg|query|src_blk>
--path <URI PATH>
--redirecttype <url_loc|url_pre|backend|use_server>
--location <URL|BACKEND>
--bool <equal|notequal>
```

Delete ACL Rules:

```
lbcli --action acl --function delete --vip <L7VIPNAME>
--pathtype <path_beg|path_end|hdr_host|hdr_beg|query|src_blk>
--path <URI PATH>
--redirecttype <url_loc|url_pre|backend|use_server>
--location <URL|BACKEND>
--bool <equal|notequal>
```

Layer 7 Header Actions

Add Header Rules:

```
lbcli --action headers --function add --vip <VIP Name>
--header_type <http-request|http-response>
--header_option <add|set|del|replace>
--header_name <X-Custom-Header>
--header_value <X-Custom-Value>
```

Delete Header Rules:

```
lbcli --action headers --function delete --vip <VIP Name>
--header_option <add|set|del|replace>
--header_name <X-Custom-Header>
```

List Header Rules:

```
lbcli --action headers --function list --vip <VIP Name>
```

Firewall Lockdown Script Actions

```
lbcli --action lockdown --enabled on --network 0.0.0.0/0
```

You turn the lockdown features 'on' and 'off' and the network is your admin subnet but if you do not wish to lockdown the management network then use ip/cidr 0.0.0.0/0.

List Actions

List floating IPs:

```
lbcli --action list --function floatingip
```

List XML as JSON:

```
lbcli --action list --function dumpconfig
```

List advanced settings:

```
lbcli --action list --function advanced --layer 4:7
List VIP for Layer4/7
lbcli --action list --function virtual --layer 4:7 --vip vipname --rip ripname
```

HA Actions

Create HA Pair:

```
lbcli --action ha_create
--local_ip <this ip>
--peer_ip <slave ip>
--peer_password loadbalancer
```

To create a HA Pair, ensure both appliances are Primary and the peer node has no configuration, no VIPs no Floating IPs and no SSL Terminations, otherwise it will fail. Once you have created a HA Pair the command will fail if run again as the peer role will be incorrect.

Hostname, DNS, Networking, Gateways and Routes Actions

Get and Set Host and Domain name:

```
lbcli --action hostname --function get
lbcli --action hostname --function set --hostname <hostname> --domain <domain_name>
```

Get and Set DNS Servers:

```
lbcli --action dns --function get
lbcli --action dns --function set --dns0 1.1.1.1 --dns1 8.8.4.4 --dns2 172.31.31.10
```

To clear a DNS Server specify which one is to be cleared with --dnsX "".

If dns0 and 1 are empty dns2 will show as dns0 in the WUI and general configuration.

Get and Set Interface MTU:

```
lbcli --action mtu --function get
lbcli --action mtu --function set --interface eth0 --mtu 1500
```

When changing the MTU, ensure the upstream device has a matching or larger MTU.

Get and Set default gateways:

```
lbcli --action route --function default --type get
lbcli --action route --function default --type set --gateway <192.168.100.1 or
2001:470:68a4::1> --interface (if not set 'Auto' is used)
```

Get and set Static Routes:

```
lbcli --action route --function static --type get
lbcli --action route --function static --type add --network <192.168.100.100/18 or
2001:470:68a4::/48> --gateway <Gateway address to get to the network>
lbcli --action route --function static --type del --network <192.168.100.100/18 or
2001:470:68a4::/48> --gateway <192.168.100.1 or 2001:470:68a4::1>
lbcli --action route --function routes --type flush
```

Flushing routes will remove all static routes.

Add and Remove IP Addresses:

```
lbcli --action address --function get|add|del|flush --interface <Interface> --address
<192.168.100.100/18 or 2001:470:68a4::10/48> --cidr <0-32 ipv4> or <64-112 ipv6>
```

get = Get addresses
add = Set a new address
del = Deletes the address specified
flush = flush an interface of all IP Addresses

Note that if --address is a simple IP without /CIDR then you need to define the --cidr. IE --cidr 24 for 255.255.255.0 netmask.

If you need assistance with netmask to cidr conversions please use the command below:

```
lbcli --help cidr
```

Policy based VIP Routing

```
lbcli --action pbr
--function <get:set:delete<
--ip <VIP IP address>
--gateway <1.2.3.4>
```

Note that --ip and --gateway are only required for --function set and delete.

if you define them for get it will return that policy if both --ip and --gateway match If no match an empty set or policies are returned.

You can only use IPv4 Floating IPs for the --ip and use of IPv6 or any other address will fail.

This allows you to define an alternate gateway for your VIPs and it used to resolve Policy Based Routing issues.

GSLB Actions

```
lbcli --action gslb --section (globalnames|members|pools|topologies) --function
(add|edit|delete|list)
```

(globalnames)

```
--name <example> --hostname <fully qualified domain name> --ttl <>
```

(members)

```
--name --ip <dns IP to publish> --monitor_ip <internal ip to monitor> --weight <0-10>
--add_member1 name --add_member2 name
```

Only valid members are accepted and not more then one member which is the same per pool entry. Two pools can share members. Members may share the backend but this will get confusing so please limit your members. Unless you need another healthcheck then redefine the same IP for a Name in an Endpoint but beware health checks may take their toll on the network. Keep health check responses minimal.

(pools)

```
--name <poolname> --monitor <http|tcp|forced|external> --monitor_interval <> --monitor_timeout
<5000> --monitor_retries <3> --monitor_use_ssl <yes:no> --monitor_status <up:down>
--monitor_hostname <fully qualified domain name / SNI Hostname> --monitor_port <80>
--monitor_expected_codes <200,201 important - no spaces>  --monitor_send_string
</example.php?status=up> --lb_method <wrr|twrr|fogroup> --fallback <any|refuse>
--max_addresses_returned <1> --add_globalname <each globalname> --add_globalname <another
globalname> --delete_globalname <delete globalname> --add_member <first members>
--delete_member <delete members>
```

```
--monitor http --monitor_use_ssl <yes:no or true:false> --monitor_hostname gslb.example.com
--monitor_url_path / --monitor_port 80 --monitor_expected_codes 200,201,301 (only 3 codes
expected)
```

```
--monitor tcp --monitor_port 80 --monitor_send_string check --monitor_match_response
```

```
--monitor forced --monitor_status <up:down>
```

```
--monitor external --monitor_script script.sh --monitor_parameters arg --monitor_result
--monitor external_dynamic_weight --monitor_port --monitor_parameters
```

To remove globalname you should use --delete_globalname and to remove members use --delete_member.

You should only use the --delete_globalname or --delete_member when editing your pools.

(topologies)

```
--name <Your AZ, DC, GEOLocaton> --add_ips <IP/CIDR> --add_ips <IP/CIDR> --add_ips <IP/CIDR>
--add_ips <Add as many IP CIDR ranges as needed> --delete_ips <IP/CIDR to remove>
```

GSLB Reports:

```
lbcli --action gslb --section reports --function get --report get_ppdns_state |
get_generic_state
```

Appliance Power Control Actions

```
lbcli --action power --function <shutdown|restart>
```

This allows you to shutdown or restart the loadbalancer.org appliance be aware that shutdown will make the appliance unresponsive and will require a visit to a hardware appliance to turn it back on.

Generate Support Archive Actions

```
lbcli --action support-download
```

This will create a support bundle in /var/www/html/tmp.

You can browse to https://<ip-of-appliance>:9443/tmp/master_YYYY-mm-dd_hh_mm_ss+0000.tar.bz2 to retrieve the file.

API Related Actions

Configure LBAPI from LBCLI:

```
lbcli --action api
--function <enable:disable|get>
--username <username>
--password <password>
--apikey <apikey>
```

When enabling the API --username and --password should be defined optionally set --apikey or a random 32 char key will be returned.

If you wish to show the API credentials then you need not specify --username --password or --apikey.

All lbcli calls can have “--method api” appended to force return output JSON, many regular calls also now return JSON.

CLI command help

for a complete list of all lbcli commands, use the following command:

```
lbcli --help lbcli
```

to obtain more detailed help for a particular action including optional sub values, use the following syntax:

```
lbcli --help <action>
```

e.g.

```
lbcli --help add-vip
```

> **Note** The CLI / API are constantly being developed, so if lbcli functionality that you require is not listed in the table above, please contact support@loadbalancer.org to check the very latest command availability.

For additional information on the CLI / API please also refer to our blog.

Running lbcli from a remote Linux Host

These commands can 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 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)

API - Version 1

Enabling the API

By default, the API is disabled. To enable the API, edit the file **/etc/loadbalancer.org/api-credentials** and uncomment the *username*, *password* and *apikey* lines, then save the file. The default username, password and apikey can be changed as required. Once enabled, API calls can be made using HTTP POST requests. As mentioned, the API enables CLI commands to be called from a Web Service.

HTTP POST Request URL

The JSON requests must be posted to the following URL on the load balancer:

```
https://<appliance IP address>:9443/api/
```

Testing

To test the functionality of the API, a browser add-on such as *HttpRequester* or *Poster* can be useful to form and post the requests.

Syntax Validation

For validating JSON syntax, the website https://jsonlint.com/ can be used. Simply paste the JSON into the window provided, then click **Validate JSON**.

Examples

To illustrate how the JSON API calls are formed, the following examples show the CLI command and the equivalent JSON API command in each case.

Example 1 - Halt a Server

This example shows how RIP1 of VIP1 is halted.

lbcli command:

```
lbcli -action halt --vip VIP1 --rip RIP1
```

JSON equivalent:

```
{
"auth": {
"apikey": "eP68pvSMM8dvn051LL4d35569d438ue0"
},
"action": [{
"command": "halt"
}],
"syntax": [{
"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:

```
{
"auth": {
"apikey": "eP68pvSMM8dvn051LL4d35569d438ue0"
},
"action": [{
"command": "add-vip"
}],
"syntax": [{
"layer": "7",
"vip": "VIP1",
"ip": "192.168.1.1",
"ports": "80",
"mode": "http"
}]
}
```

Example 3 - Add a RIP

This example shows how to add a RIP.

lbcli command:

```
lbcli --action add-rip --vip VIP1 --rip RIP1 --ip 192.168.1.2 --ports 80 --weight 100
```

JSON equivalent:

```
{
"auth": {
"apikey": "eP68pvSMM8dvn051LL4d35569d438ue0"
},
"action": [{
"command": "add-rip"
}],
"syntax": [{
"vip": "VIP1",
"rip": "RIP1",
"ip": "192.168.1.2",
"port": "80",
"weight": "100"
}]
}
```

Example 4 - Restart HAProxy

This example shows how to restart HAProxy.

lbcli command:

```
lbcli --action restart-haproxy
```

JSON equivalent:

```
{
"auth": {
"apikey": "eP68pvSMM8dvn051LL4d35569d438ue0"
},
"action": [{
"command": "restart-haproxy"
}]
}
```

Example 5 - Multiple actions in a single command

This example shows how multiple actions can be called with one POST. This example adds a layer 7 VIP, a layer 7 RIP and an STunnel VIP, then restarts HAProxy and STunnel.

```
{
"auth": {
"apikey": "eP68pvSMM8dvn051LL4d35569d438ue0"
},
"action": [{
"command": "add-vip"},
{
"command": "add-rip"},
{
"command": "termination"},
{
"command": "restart-haproxy"},
{
"command": "restart-stunnel"}
],
"syntax": [{
"layer": "7",
"vip": "VIP1",
"ip": "192.168.111.225",
"ports": "80",
"mode": "http"},
{
"vip": "VIP1",
"rip": "RIP1",
"ip": "192.168.110.240",
"port": "80",
"rip_type": "ipv4",
"weight": "100"},
{
"function": "add",
"type": "stunnel",
"vip": "SSL1",
"ip": "192.168.111.225",
"port": "443",
"backend_ip": "192.168.111.225",
"backend_port": "80",
"sslcert": "cert1"}
]
}
```

Example 6 - Using Microsoft PowerShell to call the API

This example shows how PowerShell can be used to add a layer 7 VIP.

1) Create a PowerShell file with the following contents:

PowerShell Wrapper Script for LBCLI:

```
$user = "loadbalancer"
$pass = "loadbalancer"
$ip = "192.168.111.220"
$pair = "$\{user}:$\{pass}"
$jsonfile = "c:\test-scripts\add-vip.json"
$bytes = [System.Text.Encoding]::ASCII.GetBytes($pair)
$base64 = [System.Convert]::ToBase64String($bytes)
$basicAuthValue = "Basic $base64"
$headers = @\{ Authorization = $basicAuthValue }
$json = Get-Content $jsonfile -Raw
Invoke-WebRequest -Uri "http://$\{ip}:9080/api/" -Method Post -Body $json -ContentType
"application/json" -Headers $headers
```

Note | modify $pass, $ip and $jsonfile to suit your environment

2) Create the JSON file referred to in the script: (c:\test-scripts\add-vip.json)

```
{
"auth": {
"apikey": "eP68pvSMM8dvn051LL4d35569d438ue0"
},
"action": [{
"command": "add-vip"
}],
"syntax": [{
"layer": "7",
"vip": "VIP1",
"ip": "192.168.111.228",
"ports": "80",
"mode": "http"
}]
}
```

3) Run the PowerShell script.

API - Version 2

An updated version of the API wrapper was released in v8.4.3. This release greatly simplifies the JSON layout required to configure services. The following example illustrates how the JSON layout has changed in API v2. This example adds a layer 4 DR mode VIP:

API v1 JSON layout:

```
{
"auth": {
"apikey": "eP68pvSMM8dvn051LL4d35569d438ue0"
},
"action": [{
"command": "add-vip"
}],
"syntax": [{
"layer": "7",
"vip": "VIP1",
"ip": "192.168.111.228",
"ports": "80",
"mode": "http"
}]
}
```

API v2 JSON layout:

```
{"lbcli":[
{
"action":"add-vip",
"vip":"theVipName",
"ip":"192.168.100.123",
"ports":"80",
"forwarding":"gate",
"protocol":"tcp"
}
]
}
```

As can be seen, the syntax is both simpler and easier to read.

Note

For much more information about API v2, 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 by searching for "Unix Socket Commands".

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.

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.

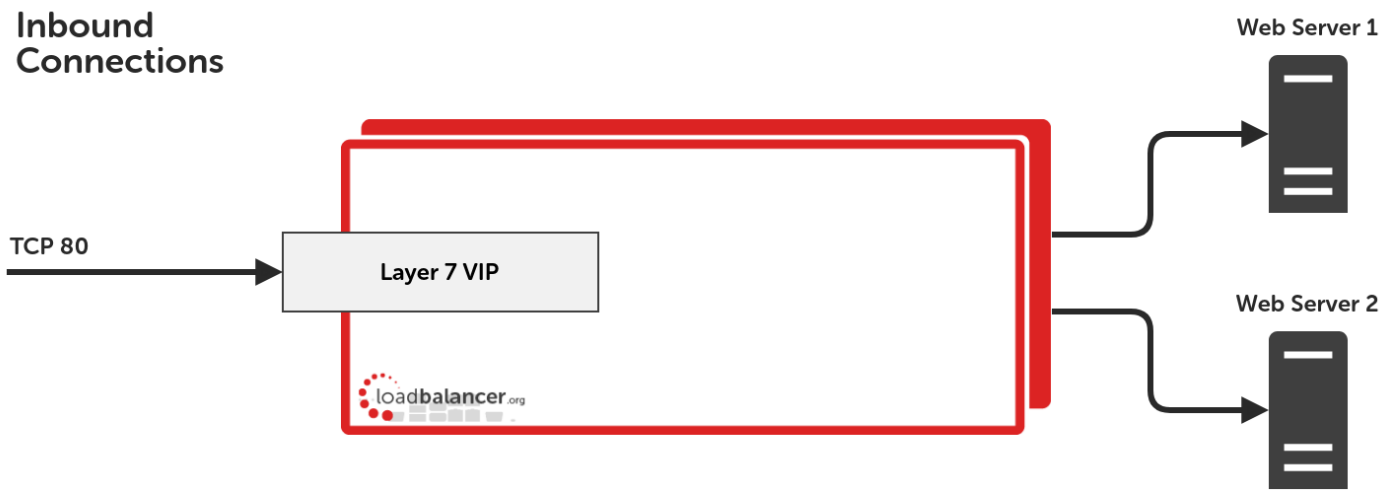
More details can be found at the OWASP Top 10 website.

Implementation Concept

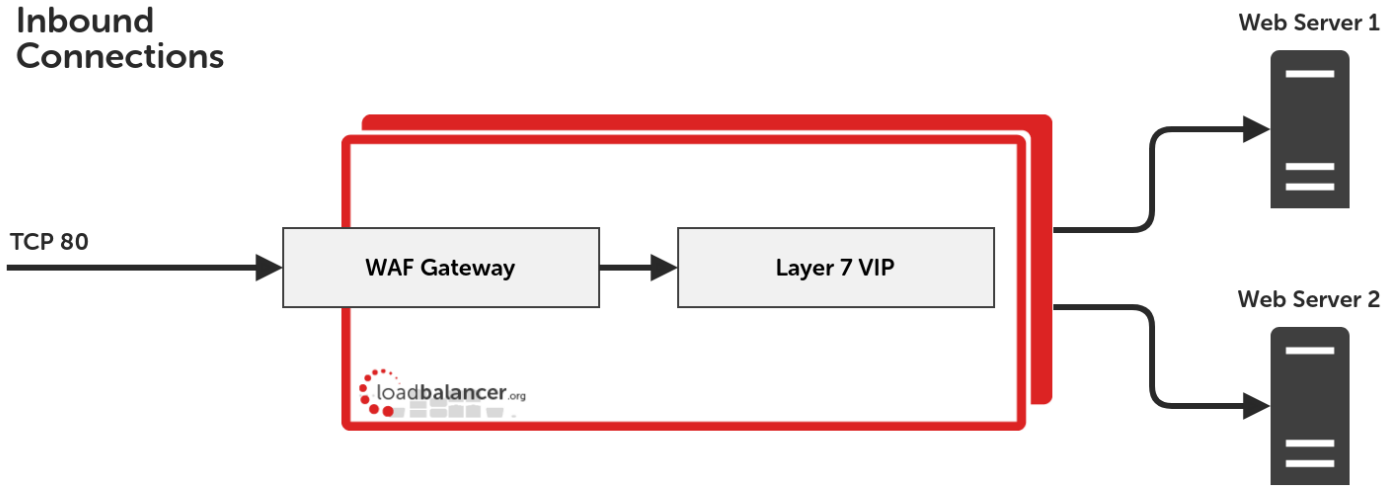
The load balancer supports the ability to define multiple WAF gateways, one for each web application to be protected. Each WAF gateway is associated with a layer 7 VIP when created. On creation, the data path is automatically modified so that the WAF becomes the initial connection point for inbound client connections, as

illustrated below:

Data flow before WAF is deployed:



Modified data flow once WAF is deployed:



Notes

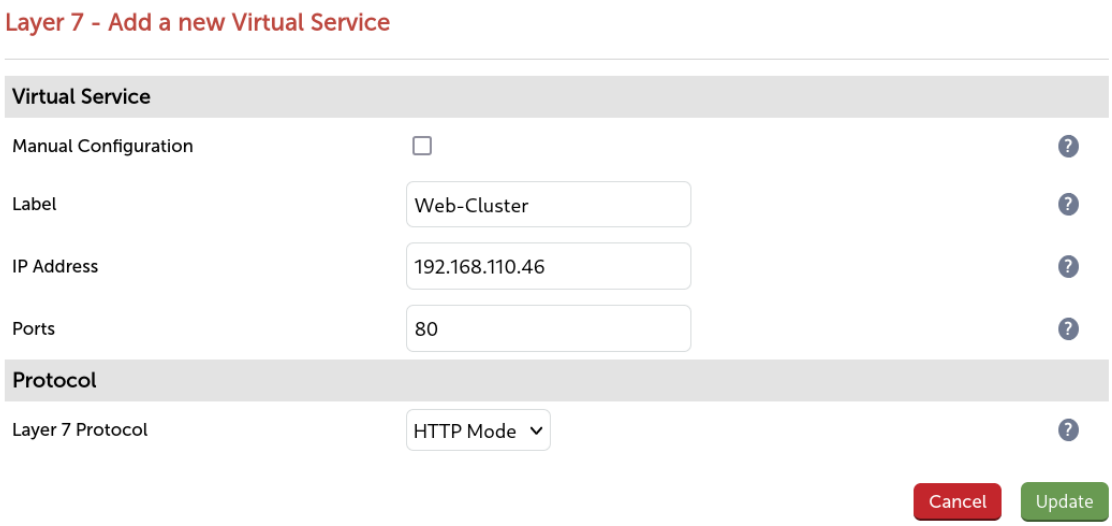
- When defining a WAF gateway on the load balancer, the associated layer 7 VIP must be selected from a drop-down list. This enables the WAF to be automatically configured to listen on the same TCP socket as the original layer 7 VIP. The WAF gateway is then automatically configured to forward packets to the original layer 7 VIP.
- Each WAF gateway is associated with one layer 7 VIP.
- Once the WAF gateway is defined, 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

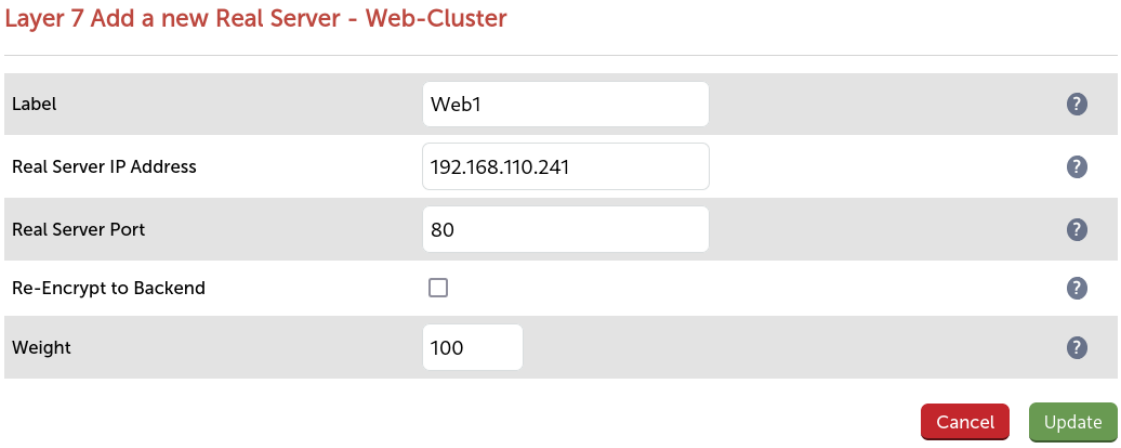
1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Virtual Services* and click **Add a new Virtual Service**.



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.

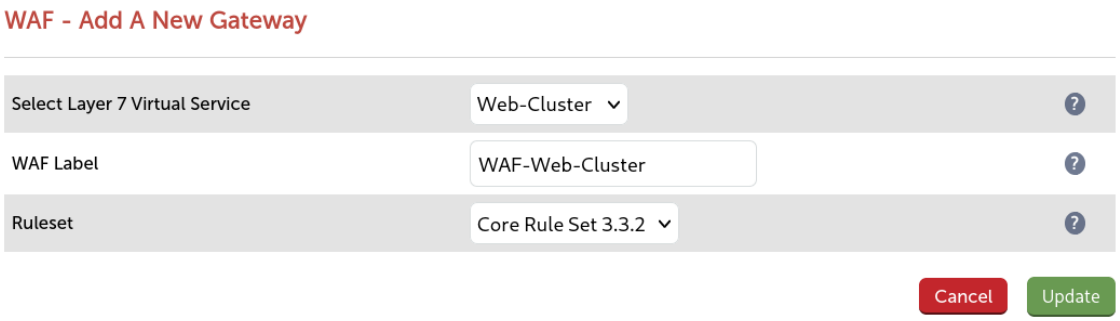


2. Enter a suitable Label (name) for the RIP, e.g. **Web1**.
3. Enter a valid IP address, e.g. **192.168.110.241**.

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**.



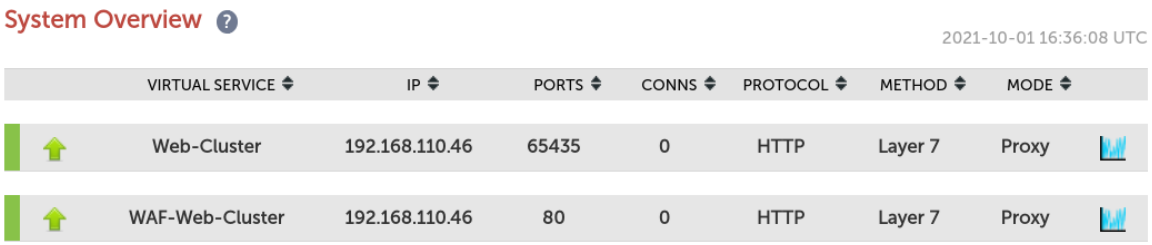
2. Select the VIP created in step 1 in the drop down.
3. The WAF label (name) field will be populated automatically, this can be changed if required.
4. The *Rule Set* will automatically choose the latest available stable version of the Core Rule Set. This should not ordinarily require changing.
5. Click **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 blue 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:



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 green **Update** button and then apply the new configuration by reloading services as prompted in the blue 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 drop-down 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

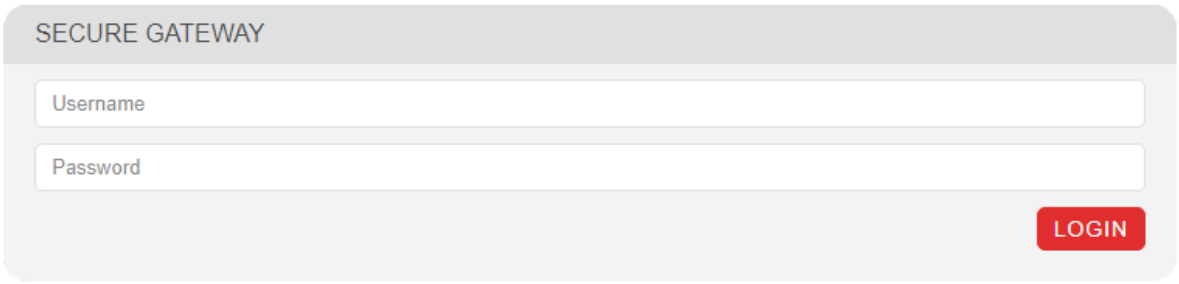
Default value: False.

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 defined static user
- Google OpenID

Once enabled, users will be prompted for credentials when accessing the WAF:



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 green **Set PCRE Match Limits** button and then apply the new configuration by reloading services as prompted in the blue message box.

PCRE Match Limit

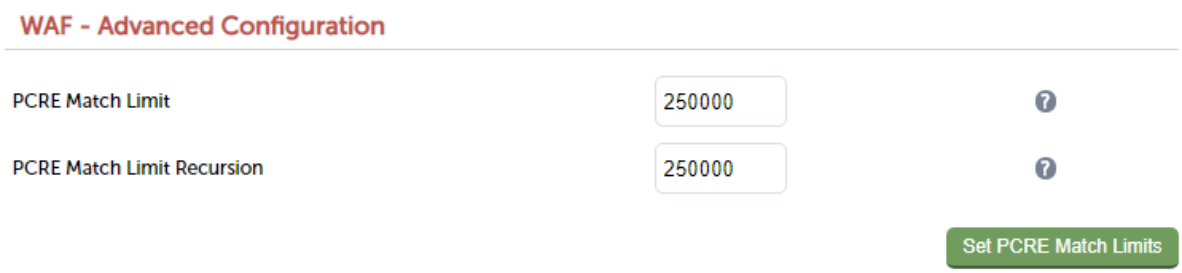
Default value: 250000.

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.



Working With the Core Rule Set

What is the Core Rule Set?

The OWASP® (Open Web 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

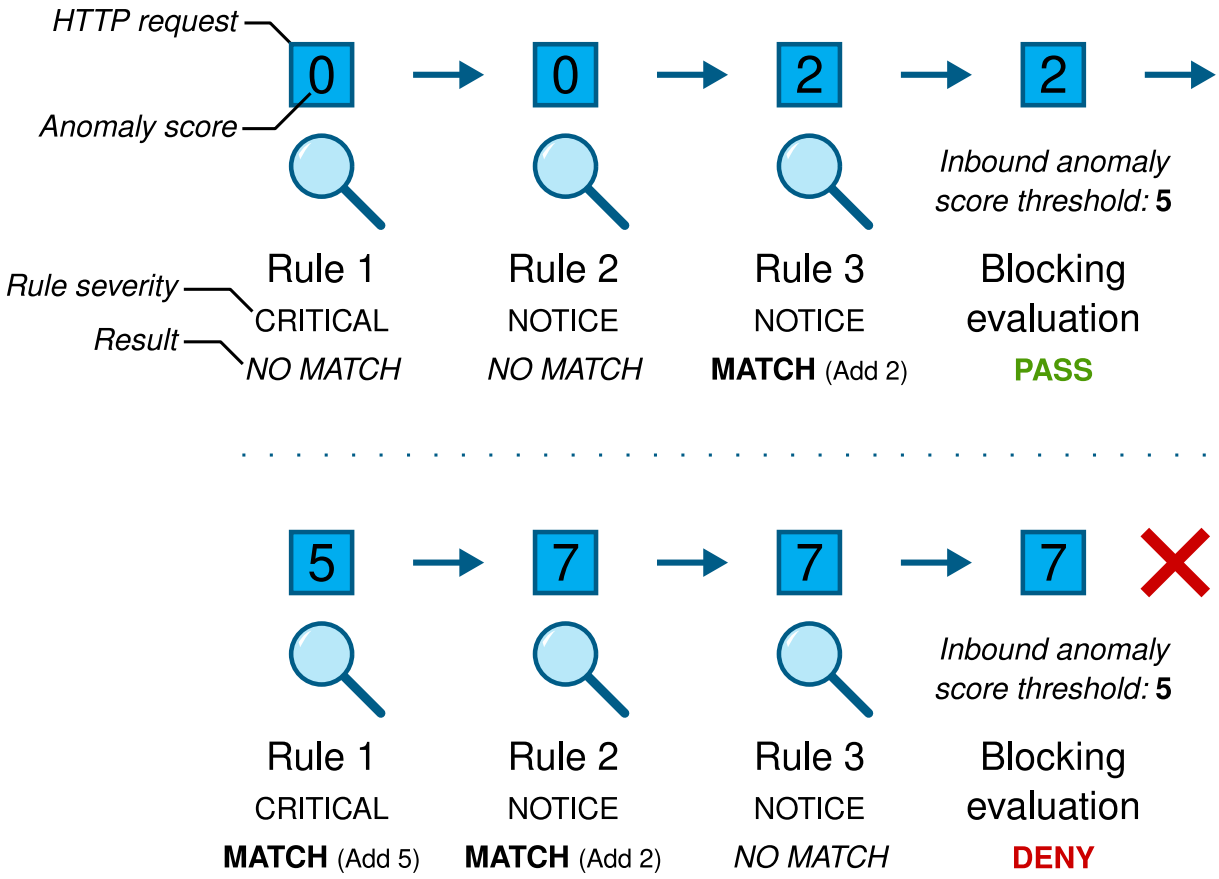
Anomaly scoring, also known as "collaborative detection", is a scoring mechanism used in the Core Rule Set. It assigns a numeric score to HTTP transactions (requests and responses), representing how 'anomalous' they appear to be. Anomaly scores can then be used to make blocking decisions. The default 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

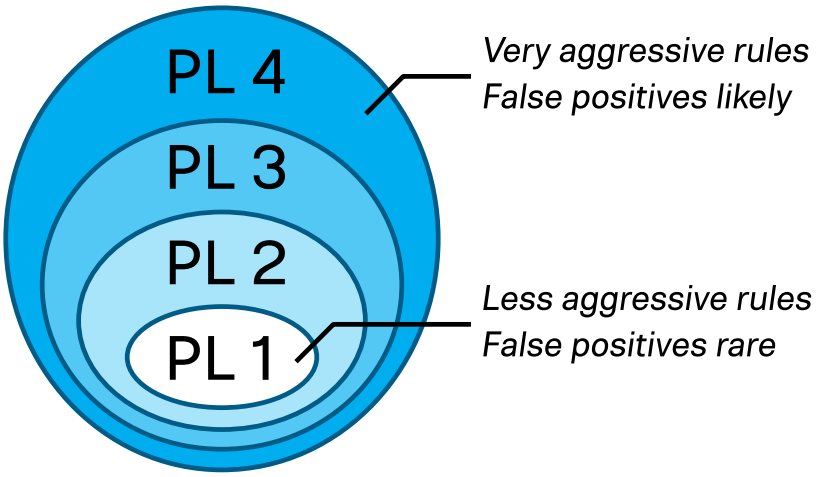
Paranoia levels are an essential concept when working with the Core Rule Set. This section explains the concept behind paranoia levels and how to work with them on a practical level.

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** drop-down list, select the desired paranoia level.

Important	After modifying any WAF gateway settings, be sure to save the new configuration to disk by pressing the green **Update** button and then apply the new configuration by reloading services as prompted in the blue 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|basefront|bdo|bgsound|big|blackface|
blink|blockquote|body|bq|br|button|caption|center|cite|code|col|colgroup|comment|dd|del|dfn|di
r|div|dl|dt|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.

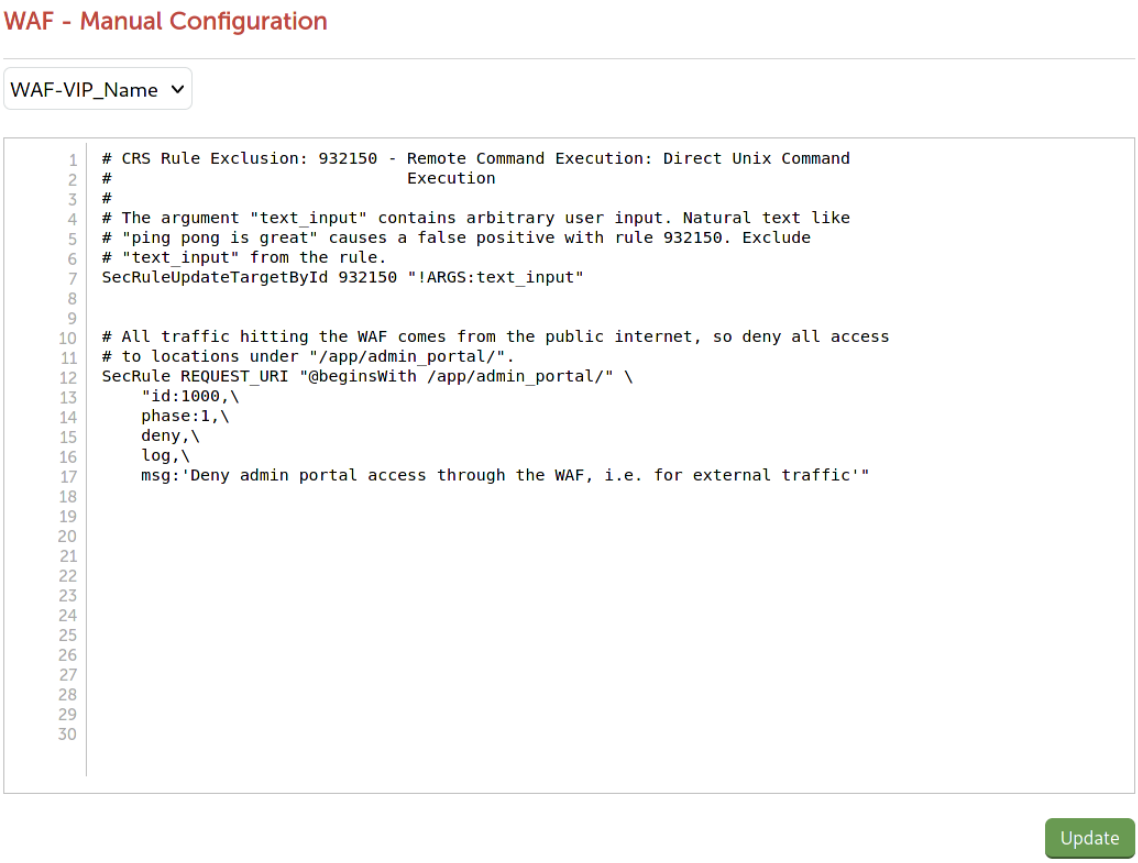
Adding Custom WAF Configuration

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 drop-down 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.



Important

After modifying any WAF manual configuration, be sure to save the new configuration to disk by pressing the green **Update** button and then apply the new configuration by reloading services as prompted in the blue message box.

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 `nolog` action can be used to prevent a rule from creating error log output and cluttering up the log.

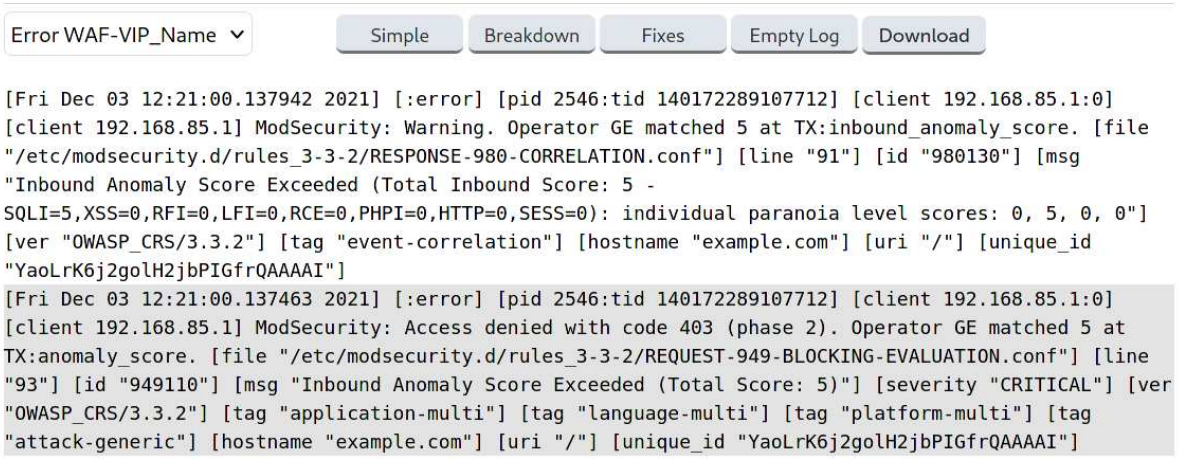
Viewing the Error Logs

To view the error logs for a given WAF gateway:

1. Using the WebUI, navigate to: *Logs > WAF Error*.
2. From the drop-down list, select the name of the WAF gateway in question.

Default View

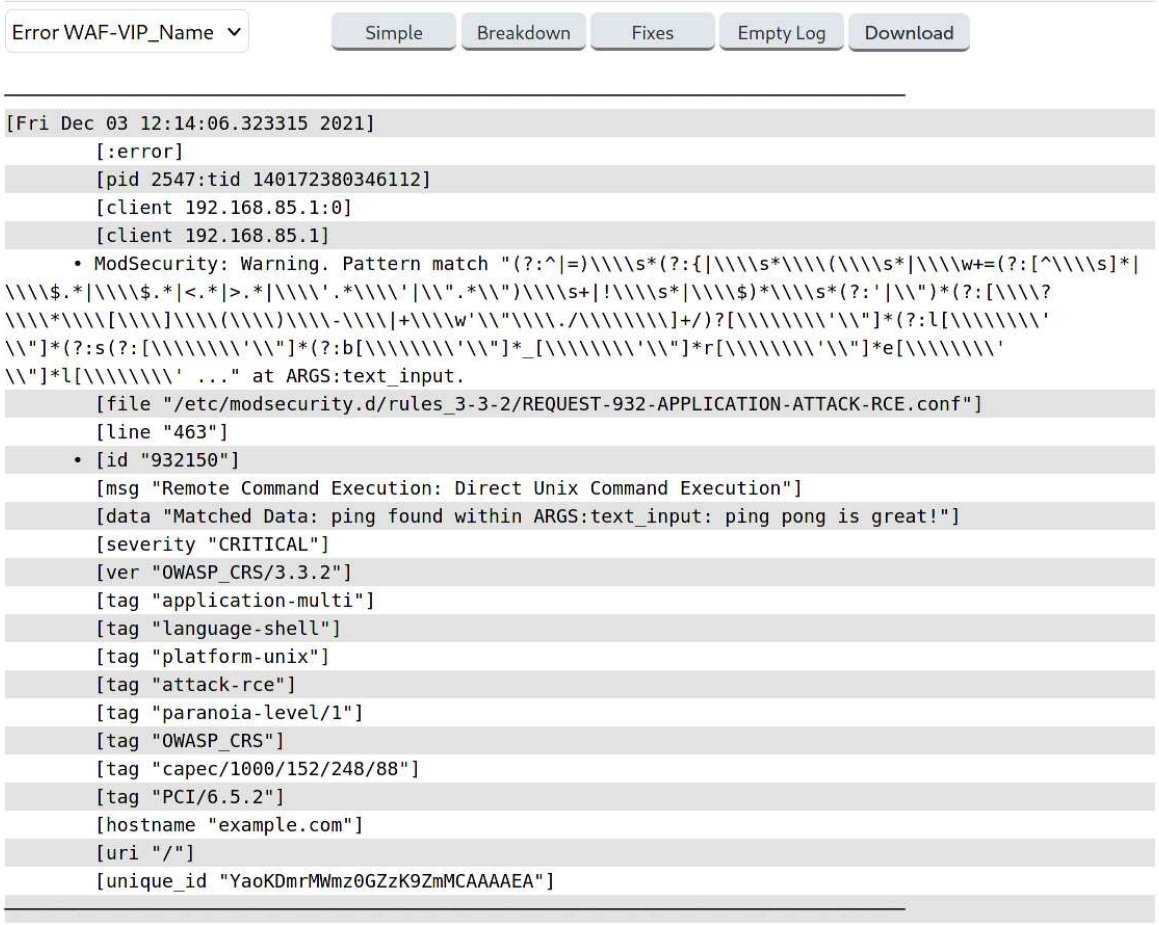
After selecting a WAF gateway from the drop-down list, the default view displays error log lines in their raw, unedited format. The most recent 500 log lines are displayed in **reverse chronological order**, with the most recent log entry shown at the top.



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.

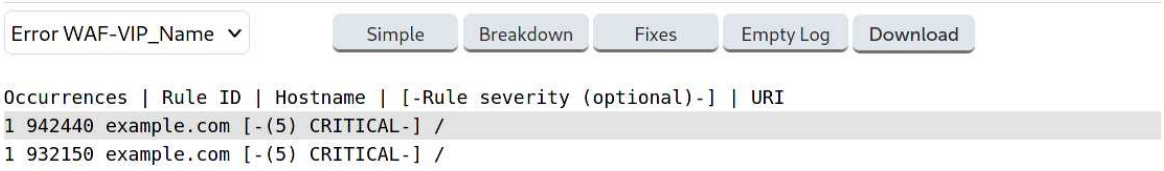
Log entries are broken up across multiple lines to aid readability. Individual log entries are separated by horizontal lines.



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.

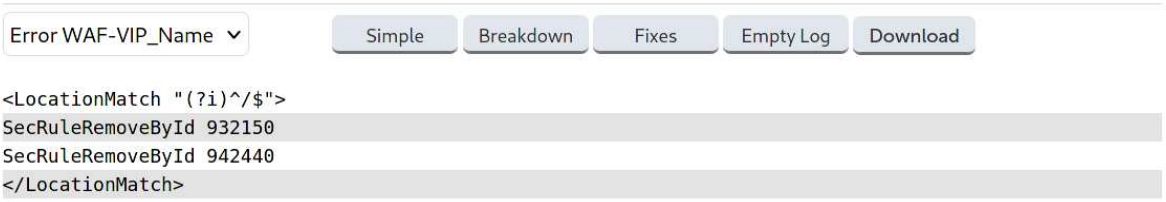


Fixes View

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.



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 "YaoKDmrMWmz0GZzK9ZmMCAAAAEA"]
```

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 "YaoKDmrMWmz0GZzK9ZmMCAAAAEA"]
```

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).

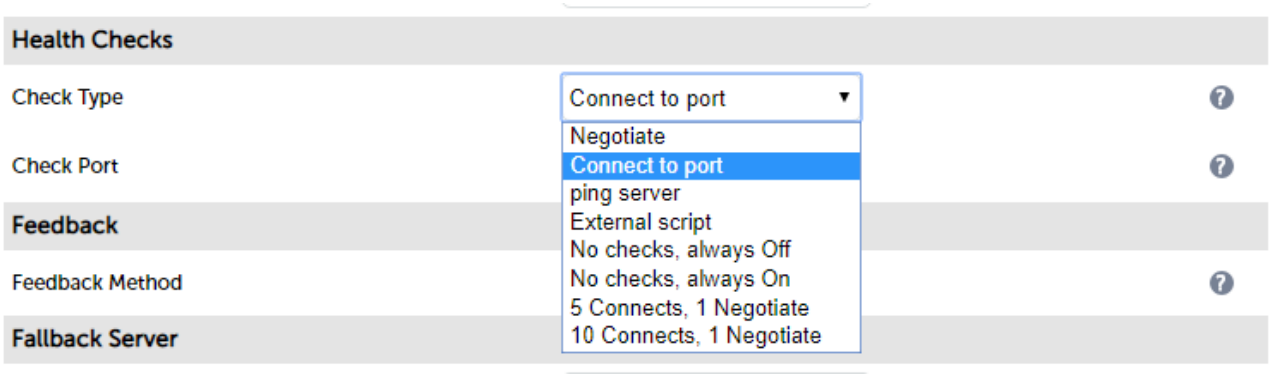
Chapter 8 - Real Server Health Monitoring & Control

Configuring Health Checks

The appliance supports a comprehensive range of health check options to check and verify the health of Real Servers. 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 heath check options available depend on the check type selected.



Note | For new Layer 4 VIPs the default check type is *Connect to Port*.

The following Check Types are supported:

Negotiate - Sends a request and looks for a specific response.

Note | If a Negotiate check is selected and *Response Expected* 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.

Connect to port - Just do a simple connect to the specified port/service & verify that it's able to accept a connection.

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 *Check Script* drop-down. For more information on using custom external health check scripts please refer to External Health Check Scripts.

No checks, always Off - All Real Servers are assumed to be down.

No checks, always On - All Real Servers are assumed to be up.

5 Connects, 1 Negotiate - Do 5 connect checks and then 1 negotiate check.

10 Connects, 1 Negotiate - Do 10 connect checks and then 1 negotiate check.

The following table describes the health check options associated with each check type:

Option	Sub Option	health check Description
Negotiate		*Send a request and matches a receive string*
	Check Port	The port to monitor. This can normally be left blank in which case the port checked is the same port defined for the Real Servers. Note that for DR mode, the port cannot be specified at the Real Server level, so the port specified for the VIP is used. Sometimes the check port differs from service port.

Option	Sub Option	health check Description
	Protocol	Set the protocol for the heath check. The options are: • **HTTP** - use HTTP as the negotiate protocol (also requires filename, path & text expected) • **HTTPS** - use HTTPS as the negotiate protocol (also requires filename, path & text expected) • **HTTP Proxy** - Use an HTTP proxy check • **FTP** - use FTP as the negotiate protocol (also requires login/password, filename in the default folder) • **IMAP (IPv4 only)** - use IMAP as the negotiate protocol (requires login/password) • **IMAPS (IPv4 only)** - use IMAPS as the negotiate protocol (requires login/password) • **POP** - use POP as the negotiate protocol (also requires login/password) • **POPS** - use POPS as the negotiate protocol (also requires login/password) • **LDAP (IPv4 only)** - use LDAP as the negotiate protocol (also requires username/password) • **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 (also requires username/password) • **Simple TCP** - Sends a request string to the server and checks the response • **RADIUS (IPv4 only)** - use RADIUS as the negotiate protocol (also requires username/password) **Additional Negotiate Check Options (depending on type selected) :** *Login* - the username when authentication is required *Password* - the password when authentication is required *Database Name* - The database to use for the MySQL check *Radius Secret* - the RADIUS secret string for the RADIUS negotiate check

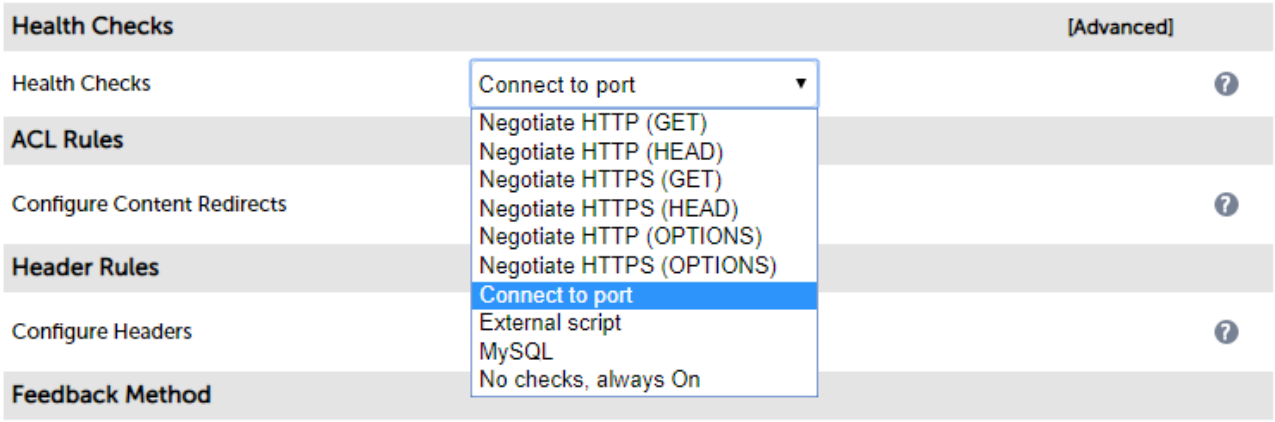
Option	Sub Option	health check Description
	Virtual Host	Used when using a negotiate check with HTTP or HTTPS. Sets the host header used in the HTTP request. In the case of HTTPS this generally needs to match the common name of the SSL certificate. If not set then the host header will be derived from the request url for the real server if present. As a last resort the IP address of the real server will be used.
	Request to Send	This is used with negotiate checks and specifies the *Request to Send* to the server. The use of this parameter varies with the protocol selected in Negotiate Check Service. 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 *Response Expected* field in not used by the SQL health check since the data returned in not read, the answer must simply be 1 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.
	Response Expected	This is the response that must be received for check to be a success. The check succeeds if the specified text (response) is found anywhere in the response from the web server when the file specified in the *Request to Send* field is requested. For example, a file called 'check.txt' could be placed in the default folder of the web server, this text file could just have the text OK in the file, then when the negotiate check runs, it would look for a file called 'check.txt' containing OK. If found, the test would succeed, if not found it would fail and no new sessions will be sent to that server. Note: If *Response Expected* 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.
Connect to Port		*Attempt to make a connection to the specified port*
	Check Port	The port to monitor. This can normally be left blank in which case the port checked is the same port defined for the Real Servers. Note that for DR mode, the port cannot be specified at the Real Server level, so the port specified for the VIP is used. Sometimes the check port differs from service port.

Option	Sub Option	health check Description
Ping Server		*Test Real Server availability using an ICMP ping*
External Script		*Use a custom file (script or binary) for the health check*
	Check Port	(See above)
	External Script	Select the required external check script from the drop-down. For more information on creating and using custom external health check scripts please refer to External Health Check Scripts. Note By default the Microsoft SQL external health check is not available in the drop down. 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: `$ lb_mssql -i` Once completed, the additional option "ms-sql-check" will appear in the External Script drop-down. 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**.
No Checks Always Off		*No checking will take place and no real or fallback servers will be activated*
No Checks Always On		*No checking will take place and Real Servers will always be assumed to be up*
5 Connects, 1 Negotiate		*Repeating pattern of 5 Connect checks followed by 1 Negotiate check*
	Check Port	(See above)
	Protocol	(See above)

Option	Sub Option	health check Description
	Virtual Host	(See above)
	Request to Send	(See above)
	Response Expected	(See above)
10 Connects, 1 Negotiate		*Repeating pattern of 10 Connect checks followed by 1 Negotiate check*
	Check Port	(See above)
	Protocol	(See above)
	Virtual Host	(See above)
	Request to Send	(See above)
	Response Expected	(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 heath 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.



Note | For new Layer 7 VIPs the default check type is *Connect to Port*.

The following Check Types are supported:

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*.

Note

If a Negotiate (Get) check is selected and *Response Expected* 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.

Connect to Port - Attempt to make a connection to the specified port.

External script - Use a custom file (typically a script but can also be a binary) for the health check. Select the script from the *Check Script* drop-down. For more information on using custom external health check scripts 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 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:

USE mysql; INSERT INTO user (Host,User) values ('','); FLUSH PRIVILEGES;

No checks, Always On - No health checks, all real servers are marked online.

Note

By default, a TCP connect health check is used for newly created layer 7 Virtual Services.

The following table describes the health check options associated with each check type:

Option	Sub Option	health check Description
Negotiate HTTP/HTTPS (GET)		*Scan the page specified in Request to Send, and check the returned data for the Response Expected string*
	Request to Send	Specify a specific location/file for the health check. Open the specified location and check for the *Response Expected*.

Option	Sub Option	health check Description
	Response Expected	The content expected for a valid health check on the specified file. The *Response Expected* can be any valid regular expression statement. For example, if the Real Servers have a virtual directory called /customers, with a default page that contains the word 'welcome', *Request to Send* would be set to "customers" (without quotes) and *Response Expected* would be set to "Welcome" (without quotes). Provided that the load balancer can access the page and see the text 'Welcome', the health check would pass. Note: If *Response Expected* 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. Note: 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 *Response Expected*.
	Advanced > Check Port	The port to monitor. This can normally be left blank in which case the port checked is the same port defined for the Real Servers. However, sometimes the check port differs from service port in which case it can be specified here. Also useful for multiport VIPs where the real server port field is left blank. In this case, the default checkport is the first in the list. This can be changed using this field if required.
	Advanced > Username	Specify a username if authentication is required.
	Advanced > Host Header	If the Real Server is configured to require a Host header, the value to be used in health checks may be set here.
	Advanced > Password	Specify a password if authentication is required.
Negotiate HTTP/HTTPS (HEAD)		*Request the page headers of the page specified in Request to Send*

Option	Sub Option	health check Description
	Request to Send	(see above)
	Advanced > Check Port	(see above)
	Advanced > Username	(see above)
	Advanced > Host Header	(see above)
	Advanced > Password	(see above)
Negotiate HTTP/HTTPS (OPTIONS)		*Request the options of the page specified in Request to Send*
	Request to Send	(see above)
	Advanced > Check Port	(see above)
	Advanced > Username	(see above)
	Advanced > Host Header	(see above)
	Advanced > Password	(see above)
Connect to port		*Attempt to make a connection to the specified port*
	Advanced > Check Port	(See above)
External Script		*Use a custom file (script or binary) for the health check*

<table>
<tr><th>Option</th><th>Sub Option</th><th>health check Description</th></tr>
<tr><td></td><td>Check Script</td><td>Select the required external check script from the drop-down. For more information on creating and using custom external health check scripts please refer to External Health Check Scripts.

Note
By default the Microsoft SQL external health check is not available in the drop down. 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:

<code>$ lb_mssql -i</code>

Once completed, the additional option "ms-sql-check" will appear in the External Script drop-down.

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.</td></tr>
<tr><td>MySQL</td><td></td><td>Check MySQL</td></tr>
<tr><td></td><td>Advanced > Username</td><td>perform a Client Authentication check, using <username> This requires an update into the MySql servers, as shown below, using MySQL client software:

USE mysql;

INSERT INTO user (Host,User) values ('<ip_of_haproxy>','<username>');

FLUSH PRIVILEGES;

Note
Without the user option, a MySql Handshake is performed</td></tr>
<tr><td>No checks, always On</td><td></td><td>No checking will take place and Real Servers will always be assumed to be up</td></tr>
</table>

External Health Check Scripts

From v8.6, custom health checks can be created and modified directly from within the WebUI. Previous versions required scripts to be created using an editor and then saved to a specific location on the appliance - this enabled

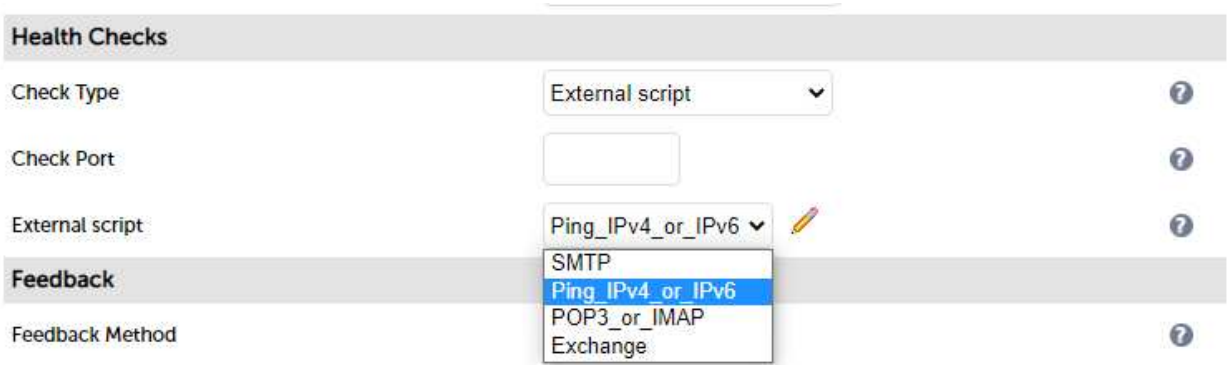
the script to be selectable when configuring external health checks.

Default Scripts

By default, 4 external scripts are available:

- SMTP
- Ping_IPv4_or_IPv6
- POP3_or_IMAP
- Exchange

These are available in the *External Script* drop-down when the *Check Type* for a VIP is set to **External Script** as shown below:



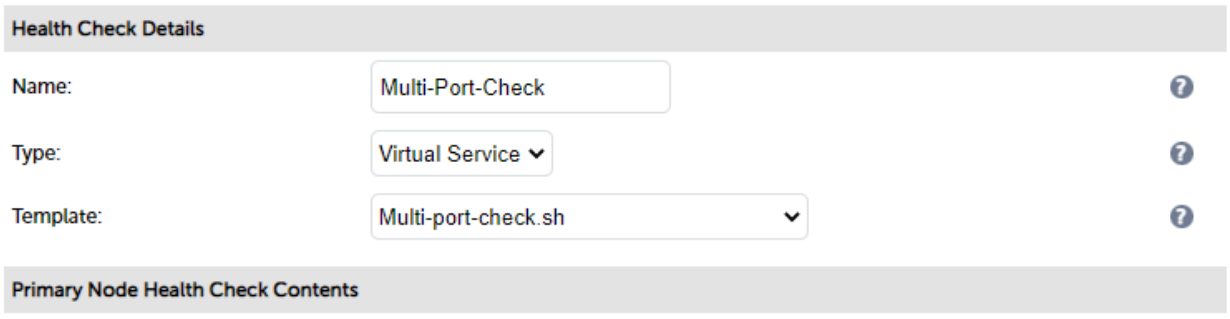
Adding Additional Health Check Scripts

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:

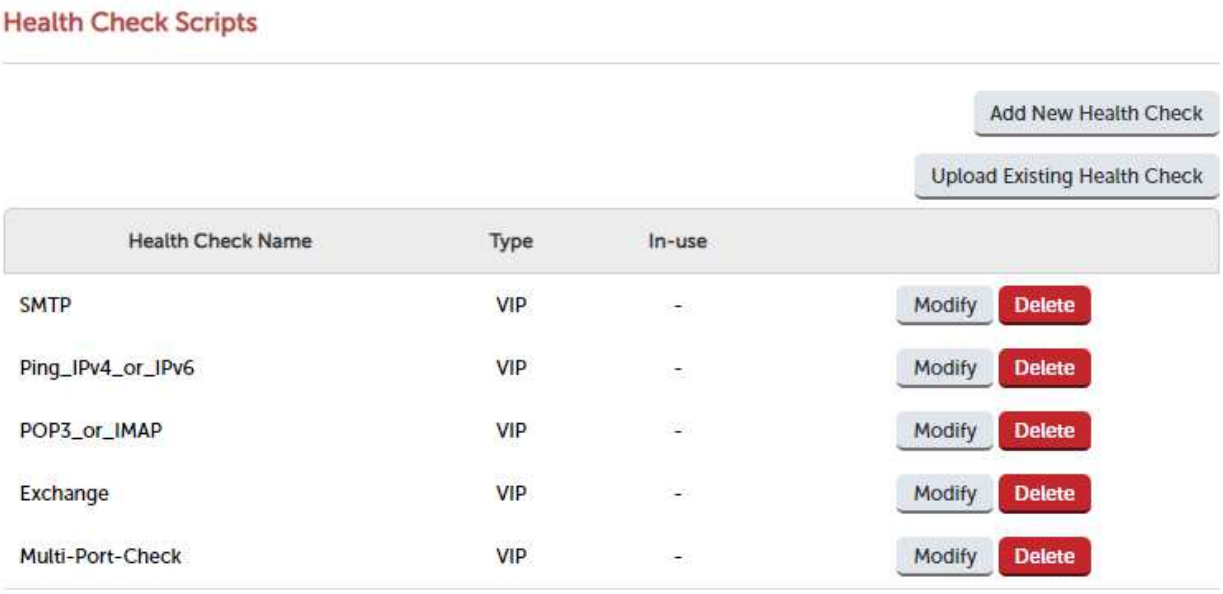
1. Using the WebUI, navigate to *Cluster Configuration > Health Check Scripts* and click **Add New Health Check**.



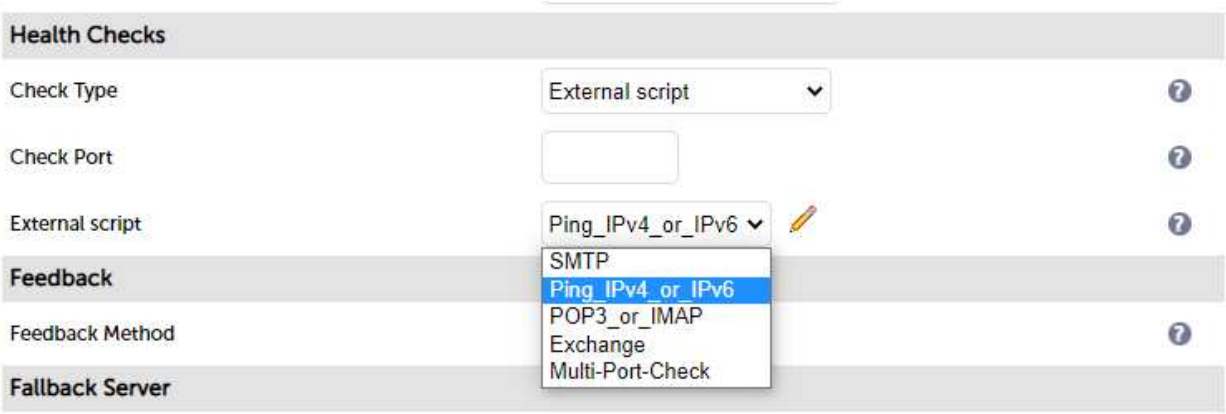
2. Specify an appropriate *Name* for the health check, e.g. **Multi-Port-Check**.
3. Set *Type* to **Virtual Service**.
4. Using the *Template* dropdown select an appropriate template from the **Virtual Service** section of the list, e.g. **Multi-port-check.sh**.
5. Modify the script if required.

6. Click **Update**.

Once the health check has been added, it will appear in the Health Check Scripts list below the 4 default scripts as shown below:



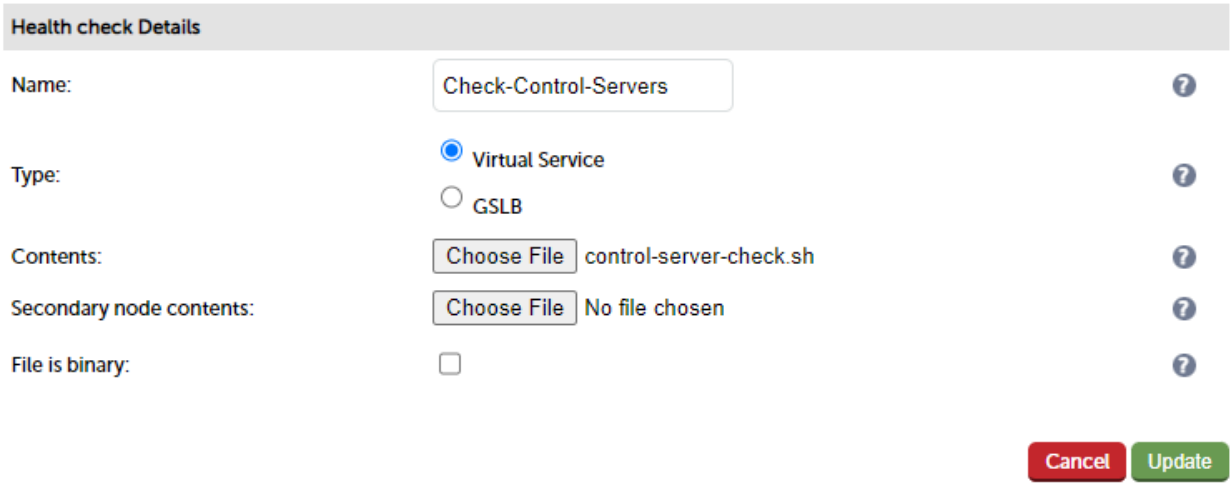
The new script will also appear in the *External Script* dropdown when the *Check Type* for a VIP is set to **External Script**:



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**.



2. Specify an appropriate *Name* for the health check, e.g. **Check-Control-Servers**.
3. Set *Type* to **Virtual Service**.
4. Click the **Choose File** button next to *Contents*.
5. 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.

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 *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 option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, enable 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 these rules are cleared 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. The 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 via the WebUI

The fallback page can be provided in the following ways:

- Using the load balancer's built in NGINX 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

```
<html>
<head>
<title>The page is temporarily unavailable</title>
<style>
body { font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body bgcolor="white" text="black">
<table width="100%" height="100%">
<tr>
<td align="center" valign="middle">
The page you are looking for is temporarily unavailable.<br/>
Please try again later.<br/>
(WUI port reminder 9080)
</td>
</tr>
</table>
</body>
</html>

```

Notes

1. The local fallback server is an NGINX instance that by default listens on port 9081.
2. If a layer 4 VIP is added that listens on port 80, NGINX is automatically configured to listen on ports 9081 & 80.
3. You can use any valid HTML for the default page, simply copy and paste the required HTML into the Fallback Page.
4. 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 change the fallback server to be one of your internal servers.

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, don't forget to solve the ***ARP Problem*** for the dedicated fallback server. For more information please refer to The ARP Problem.

For NAT mode don't forget to set the default gateway of the fallback server to the internal IP of the load balancer or when you have 2 appliances in a cluster, to a floating IP.

Using a Layer 7 VIP

It's 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

Email alerts can be configured for Virtual Services. This enables emails to be sent when Real Servers fail their health checks and are removed from the table, and also when they subsequently start to pass checks and are re-added to the table.

Layer 4

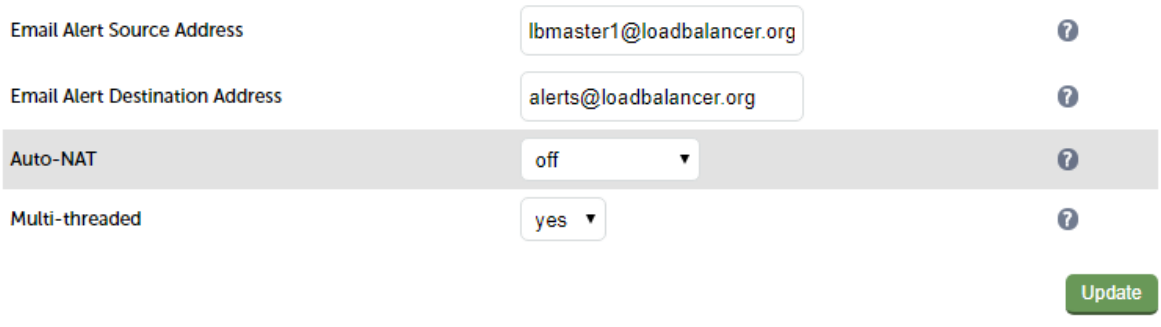
At layer 4, settings can be configured globally that apply to all VIPs or individually to each VIP.

Global 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*.



2. Enter an appropriate email address in the *Email Alert Source Address* field.

```
e.g. lbmaster1@loadbalancer.org
```

3. Enter an appropriate email address in the *Email Alert Destination Address* field.

```
e.g. alerts@loadbalancer.org
```

4. Click **Update**.

Note

Make sure that you also configure an SMTP smart host using the WebUI menu option: *Local Configuration > Physical Advanced configuration > Smart Host*. This will be auto-configured (if a DNS server has already been defined) to the MX record of the destination address domain name.

VIP Level Settings

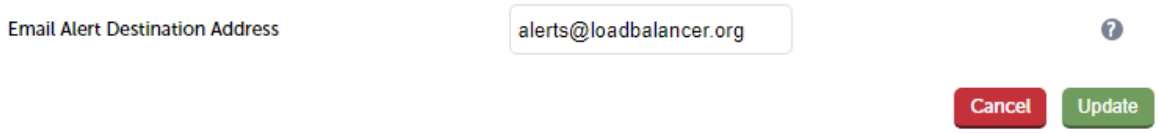
Once configured, these settings apply to individual VIPs.

To configure VIP level email alerts:

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 4 Advanced Configuration*.
2. Enter an appropriate email address in the *Email Alert Source Address* field.

```
e.g. LB1@loadbalancer.org
```

3. Using the WebUI, navigate to: *Cluster Configuration > Layer 4 Virtual Service* and click **Modify** next to the VIP to be configured.
4. Scroll down to the *Fallback Server* section.



5. Enter an appropriate email address in the *Email Alert Destination Address* field.

```
e.g. alerts@loadbalancer.org
```

6. Click **Update**.

Note

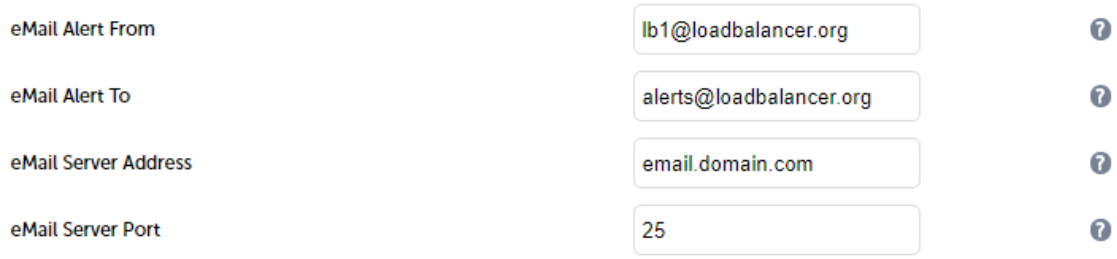
Make sure that you also configure an SMTP smart host using the WebUI menu option: *Local Configuration > Physical Advanced configuration > Smart Host*. This will be auto-configured (if a DNS server has already been defined) to the MX record of the destination address domain name.

Layer 7

At layer 7, email settings must be configured globally rather than at the individual VIP level.

To configure global email alert settings for layer 7 services:

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 Advanced Configuration*.



2. Enter an appropriate email address in the *eMail Alert From* field.

```
e.g. lbmaster1@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. email.domain.com
```

5. Enter an appropriate port in the *eMail Server Port* field.

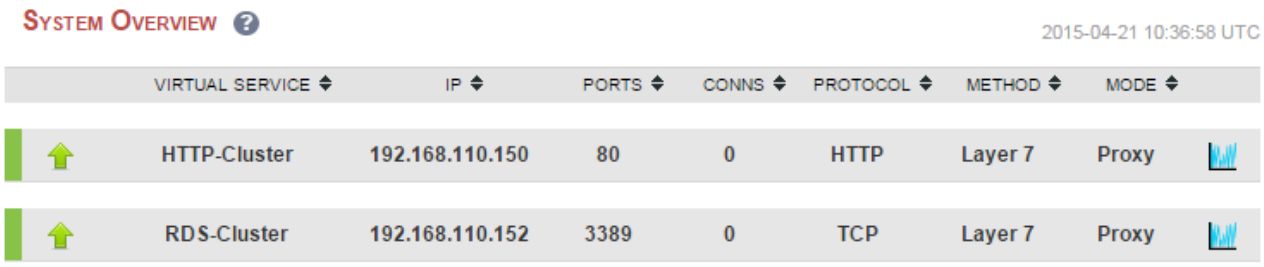
```
e.g. 25
```

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:

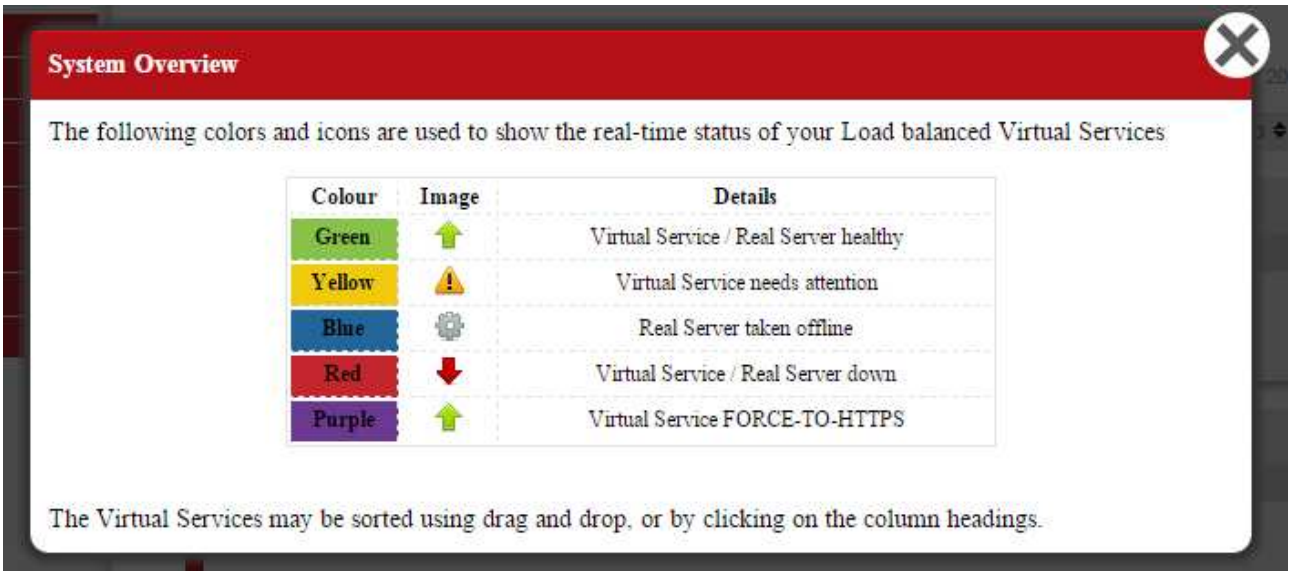


Clicking on each Virtual Service expands the view so that the associated Real Servers can also be seen:

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:

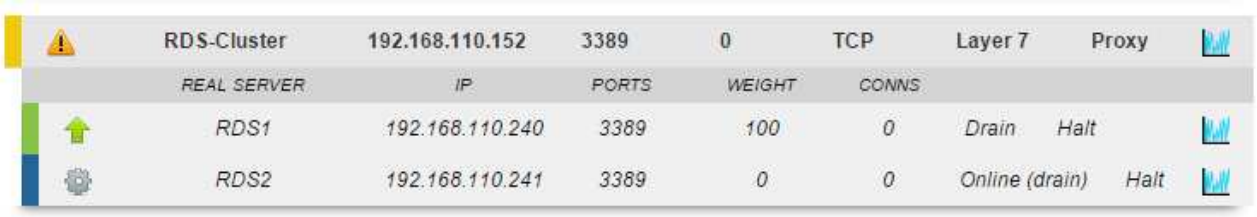


Real Server Control

The System Overview also enables each Real Server to be taken offline. This can be achieved in 2 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 immeadiately without waiting. It also prevents any new connections being established.

The screen shot below shows that RDS2 has been placed in drain mode:



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.

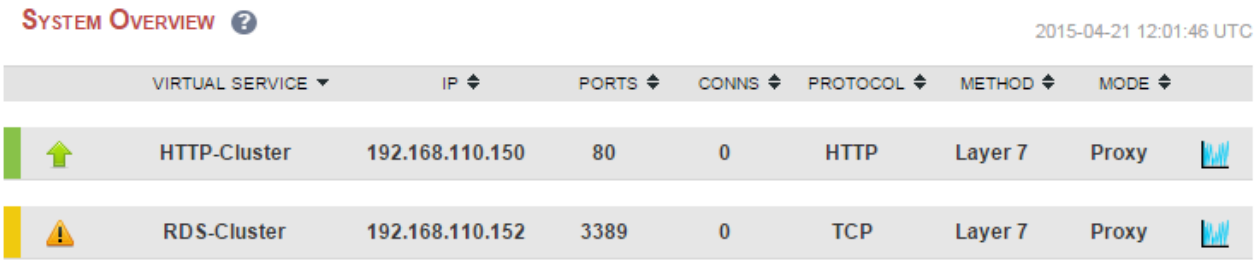
Note	Halting or draining all Real Servers in a cluster activates the fallback server.

Ordering of VIPs

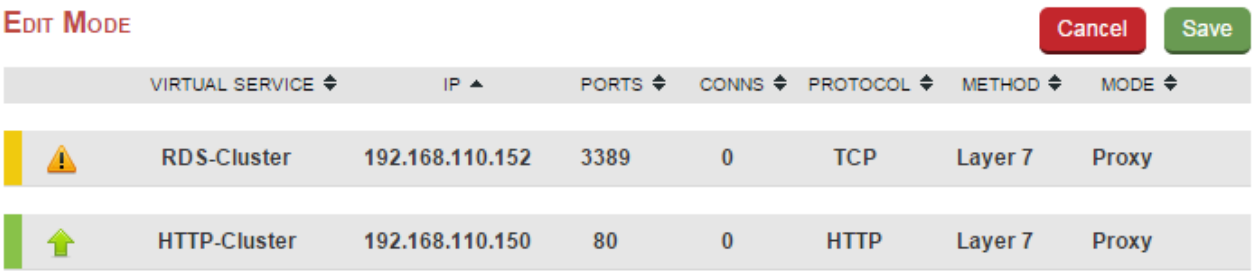
The display order of configured VIPs can be changed either by clicking on the column heading, or by drag and drop.

Sort by Column

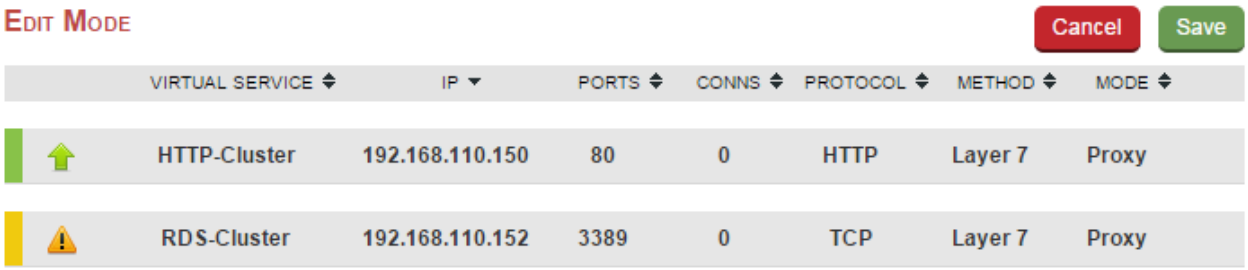
If VIPs are ordered by a particular column, this is indicated using arrows next to the column heading as shown below:



In this example, the VIPs are ordered alpha-numerically by Virtual Service name. 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 shows the following:



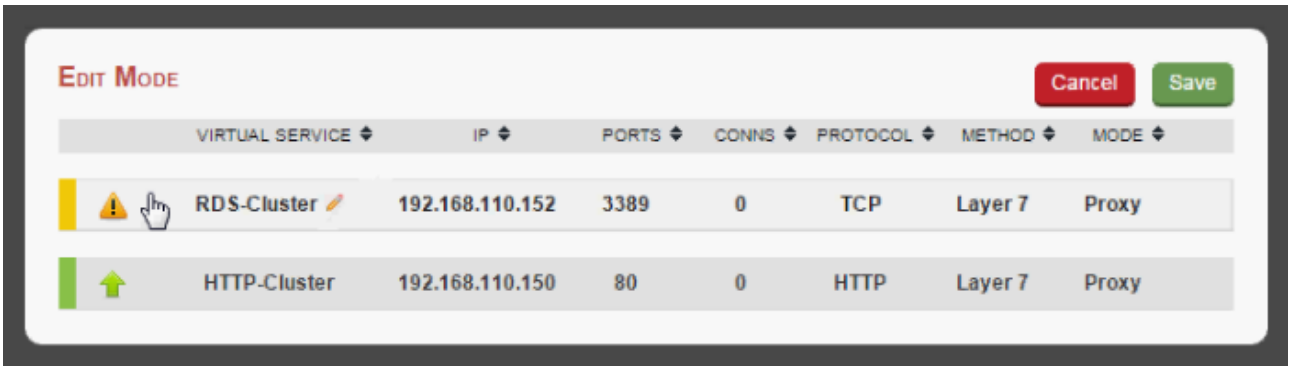
Clicking on the IP column heading again changes the order to:



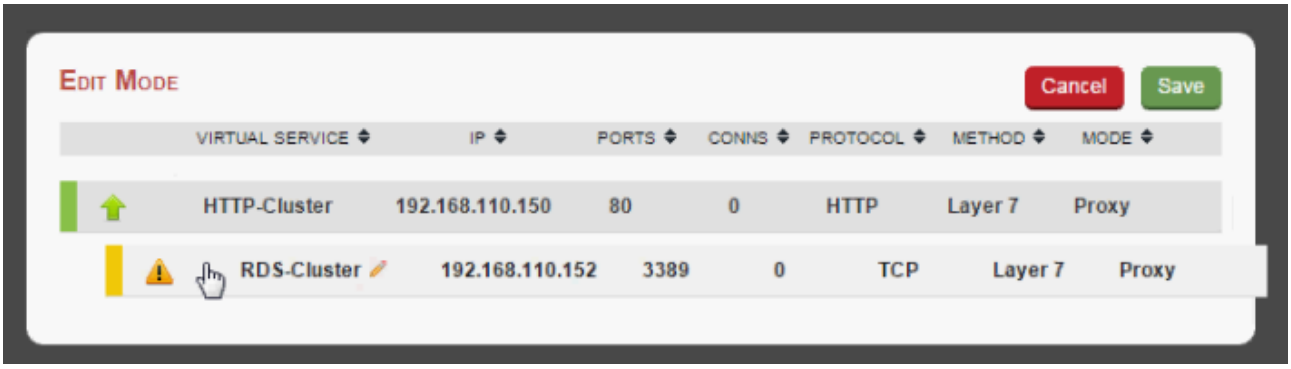
Once you've set the required order, click **Save**.

Drag & Drop

To re-order VIPs by drag and drop, simply click the mouse on any part of the VIP:



Then drag it to the required position:

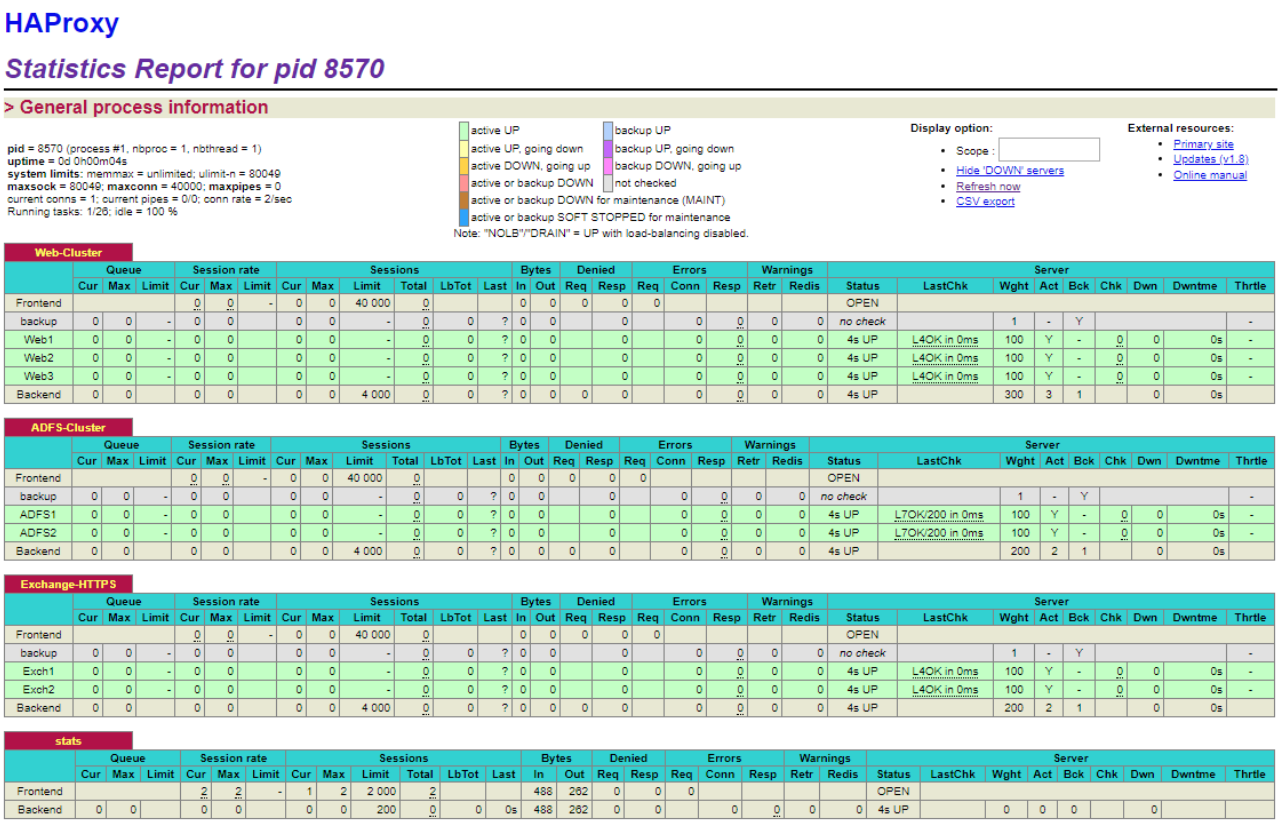


And release it. Once you've set the required order, click **Save**.

Real Server Monitoring & Control using the HAProxy Statistics Page

Real Server Monitoring

The layer 7 Statistics Page includes a visual display indicating the health status of all Virtual and Real Servers as shown in the example below:



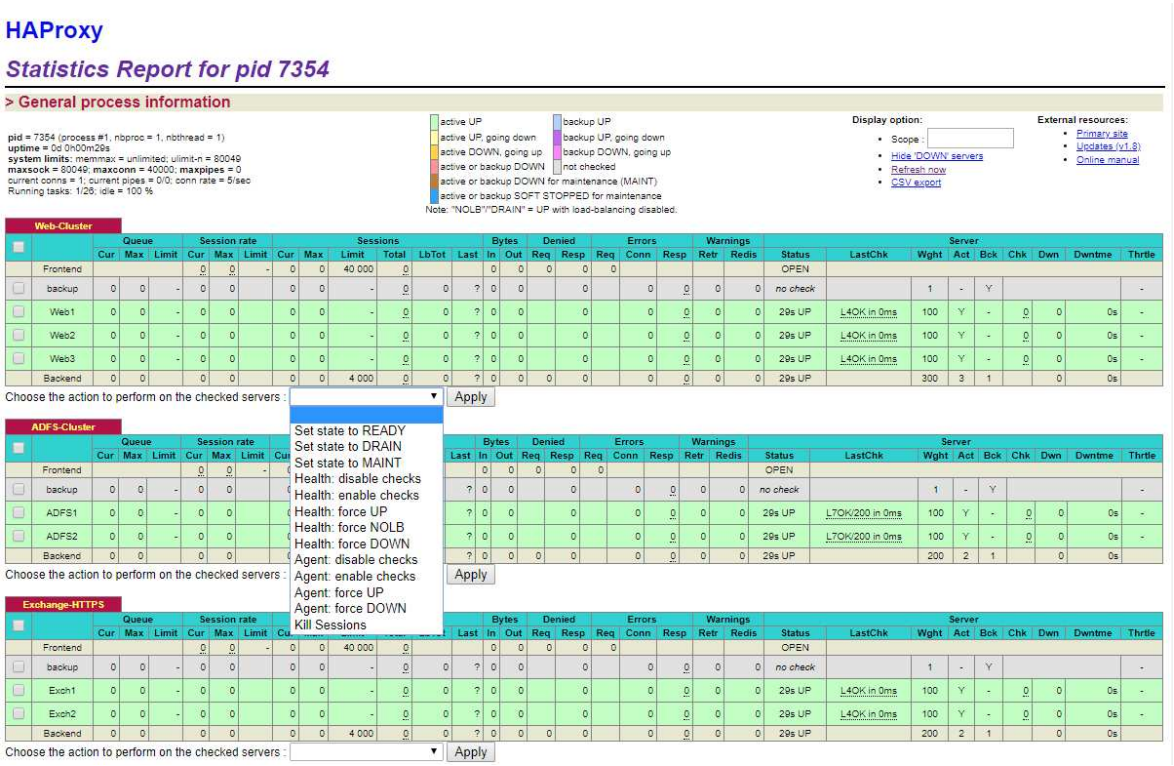
To access the page, navigate to: *Reports > Layer 7 Status* - a new tabbed window will be opened.

Real Server Control

It's also possible to control layer 7 Real Servers using the HAProxy statistics page. 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:



5. Use the checkboxes to select the relevant Real Server(s), then select the required action in the drop-down, 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 unit (normally the Primary) handles all traffic under normal circumstances. If the active unit fails, the passive unit (normally the Secondary) becomes active and handles all traffic.

We recommend that the Primary appliance is fully configured first, then the Secondary should be added. Once the Primary and Secondary are paired, all load balanced services configured on the Primary are automatically replicated to the Secondary over the network using SSH/SCP. Pairing must be performed on the unit 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, 2 private IPs must be specified – one for the VIP when it's active on the Primary and one 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
- Communicate with each other on UDP port 6694 (or the selected custom port if this has been changed)

Heartbeat

By default, heartbeat uses ucast over 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 unit if the active unit 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*. When serial communication is disabled, console access via the serial port is activated.

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 settings related to the layer 4 and layer 7 load balanced services are automatically replicated from Primary to Secondary. This ensures that should the Primary unit fail, the Secondary is already configured to run the same services. Note that 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

Settings that are not replicated and therefore must be manually configured on the Secondary unit are listed by WebUI menu option in the table below:

WebUI Main Menu Option	Sub Menu Option	Description
Local Configuration	Hostname & DNS	Hostname and DNS settings
Local Configuration	Network Interface Configuration	All network settings including IP address(es), bonding configuration and VLANs
Local Configuration	Routing	Routing configuration including default gateways and static routes
Local Configuration	System Date & time	All time and date related settings
Local Configuration	Physical – Advanced Configuration	Various settings including Internet Proxy, Management Gateway, Firewall connection tracking table size, NIC offloading, SMTP relay, logging and Syslog Server
Local Configuration	Security	Appliance security settings
Local Configuration	SNMP Configuration	Appliance SNMP settings
Local Configuration	Graphing	Appliance graphing settings
Local Configuration	License Key	Appliance licensing
Maintenance	Software Updates	Appliance software update management
Maintenance	Firewall Script	Appliance firewall (iptables) configuration
Maintenance	Firewall Lockdown Wizard	Appliance management lockdown settings

Manually Forcing Appliance Synchronization

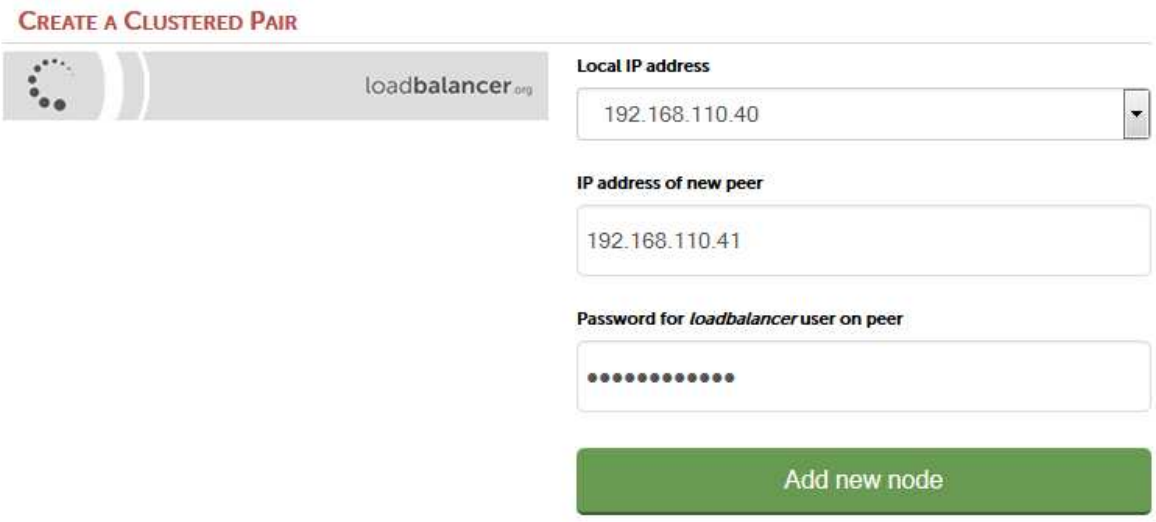
To Force Primary to Secondary Synchronisation:

1. Using the WebUI, navigate to: *Maintenance > Backup & Restore*.
2. Select the Synchronization tab.
3. Click **Synchronize Configuration with peer**.

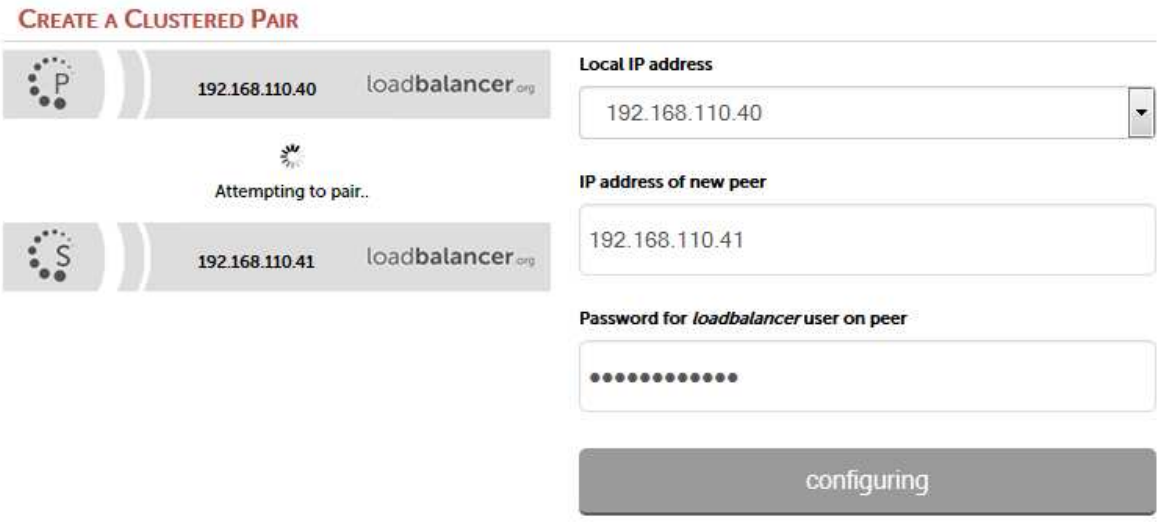
Synchronize Configuration with peer - replicate the load balanced services configuration from the Primary to the Secondary device.

To Create an HA Pair (Add a Secondary)

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*.

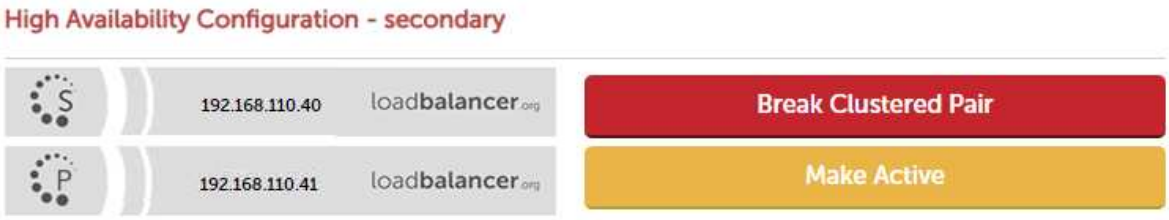


3. Specify the IP address and the *loadbalancer* users password (the default is 'loadbalancer') for the Secondary (peer) appliance as shown above.
4. Click **Add new node**.
5. A warning will be displayed indicating that the pairing process will overwrite the new Secondary appliance's existing configuration, click **OK** to continue.
6. The pairing process now commences as shown below:



7. Once complete, the following will be displayed under: *Cluster Configuration > High Availabuility Configuration* on the Primary appliance:

The following will be displayed under: *Cluster Configuration > High Availabuility Configuration* on the Secondary appliance:

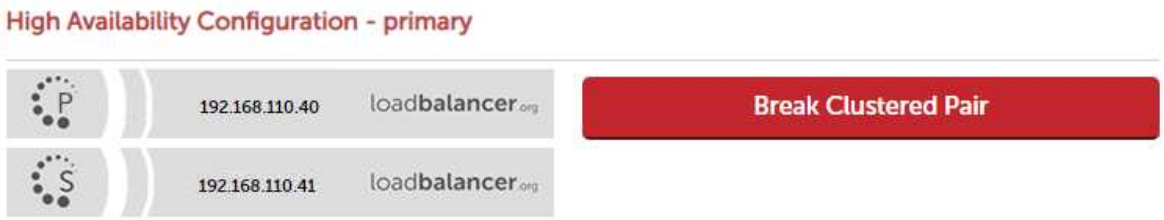


8. Now restart heartbeat as prompted in the blue message box at the top of the screen.

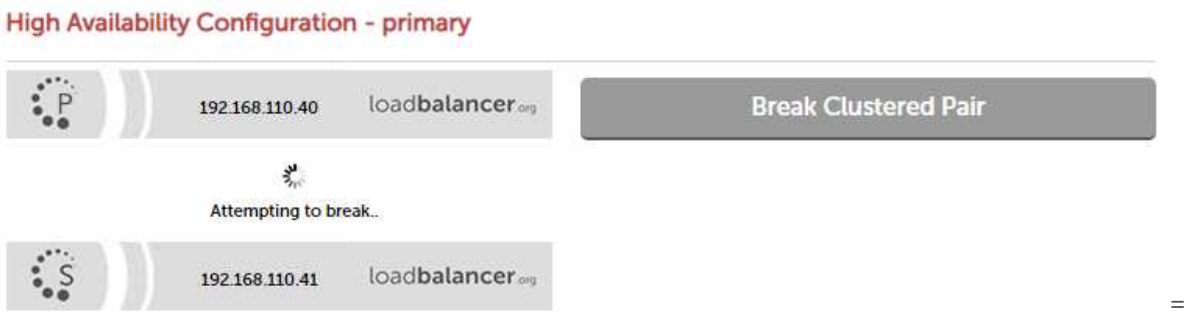
Note	Clicking the **Restart Heartbeat** button on the Primary appliance will also automatically restart heartbeat on the Secondary appliance.

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).

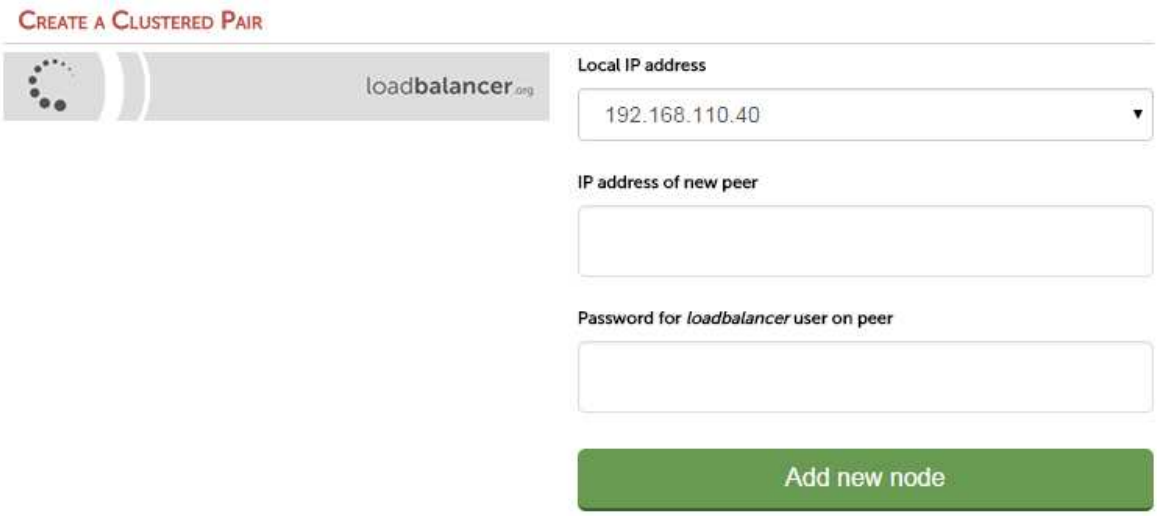


2. To break the pair, click the red **Break Clustered Pair** button.
3. Click **OK** to confirm you want to proceed.



=

4. Once the process is complete, the pairing configuration screen will be displayed on both appliances:



5. To complete the reconfiguration, restart the system services on both appliances as directed in the blue message box.

Notes

1. Load balanced services will be momentarily interrupted as system services are restarted.
2. After the pair is broken, the Secondary 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.

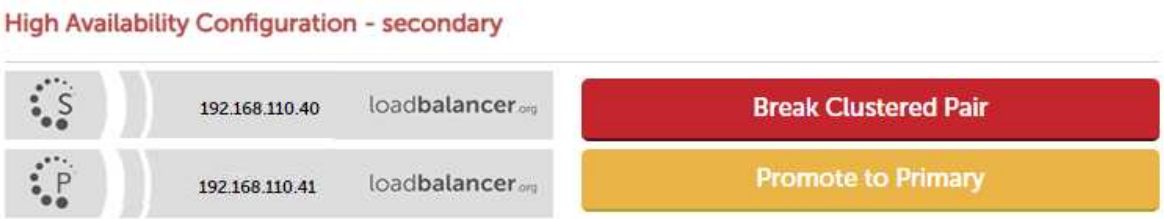
Alternatively, use the WebUI menu option: *Maintenance > Backup & Restore > Restore > Restore Manufacturer's Defaults* to clear all settings and return to default settings.

Promoting a Secondary to Primary

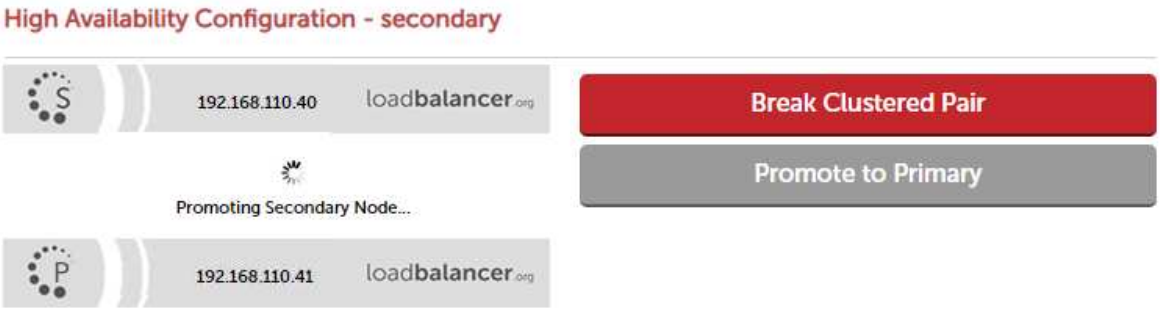
This is useful if the Primary unit fails and you'd like to change the now active Secondary to be a Primary, and then add the repaired/replaced Primary back as a Secondary unit.

To promote a Secondary unit to become a Primary:

1. Using the WebUI of the Secondary appliance, navigate to: *Cluster Configuration > High-Availability Configuration*.



2. Click **Promote**.
3. Click **OK** to confirm you want to proceed.



Note | If the Primary is still up and operational, it will not be possible to promote the Secondary.

4. Once complete, the unit will be configured as a Primary.

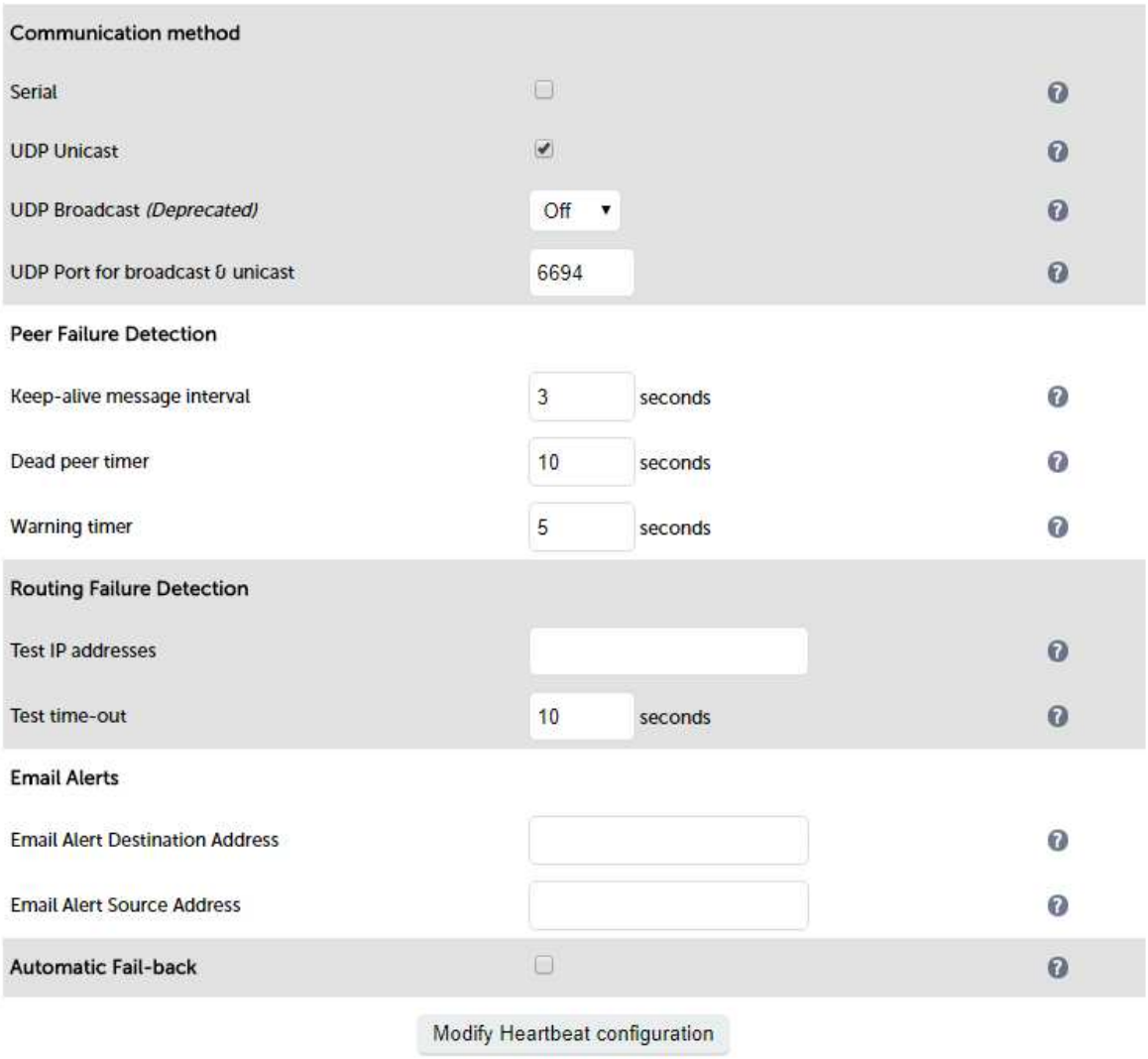
Note | Please refer to Chapter 15 - Backup & Restore and Disaster Recovery for more details on how to recover from various appliance failure scenarios.

Configuring Heartbeat

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.



Serial - Enable or disable heartbeat Primary/Secondary communication over the serial port. 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 (one cable is supplied with each appliance) to be connected between the two load balancers in the cluster. This enables heartbeat checks to utilize the serial port. When serial communication is disabled, console access via the serial port is activated.

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.

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.

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 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.

Automatic Fail-back - Enable/disable auto-failback. When the Primary returns to service after a failure, should it become active again? 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 unit to return to passive mode. For more information on controlling failback please refer to Forcing Primary/Secondary Failover & Failback.

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) swaps 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 using SSH or at the console, then as root run the following commands:

```
ipvsadm --start-daemon master
ipvsadm --start-daemon backup
```

Next, login to the Secondary appliance using SSH or at the console, then as root 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 option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, enable 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 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 persistent connection table to work when the Primary load balancer swaps over to the Secondary then this can be enabled using the WebUI. Enabling this option will replicate persistence tables for all relevant layer 7 VIPs to the peer load balancer.

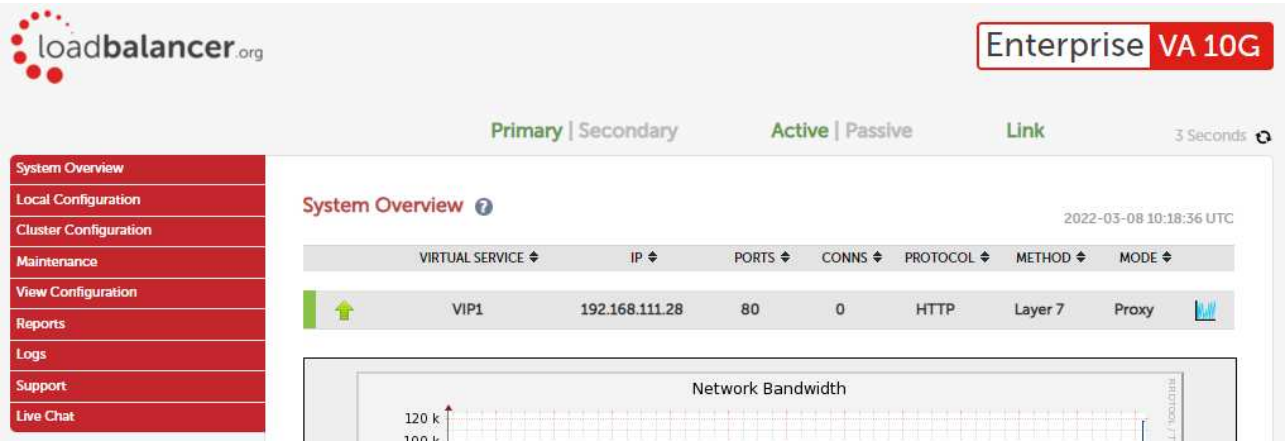
To enable persistence state table replication:

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 blue box at the top of the screen.

Clustered Pair Diagnostics

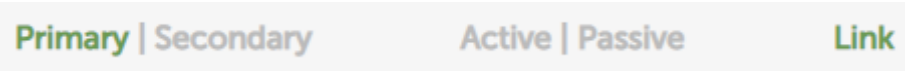
Heartbeat State Diagnostics

The status of the appliance is shown at the top of the screen. For a working pair, the normal view is shown below:



This shows that the Primary unit is active and that the heartbeat link is up between Primary & Secondary.

If no VIPs are defined, the status on Primary & Secondary appear as follows:

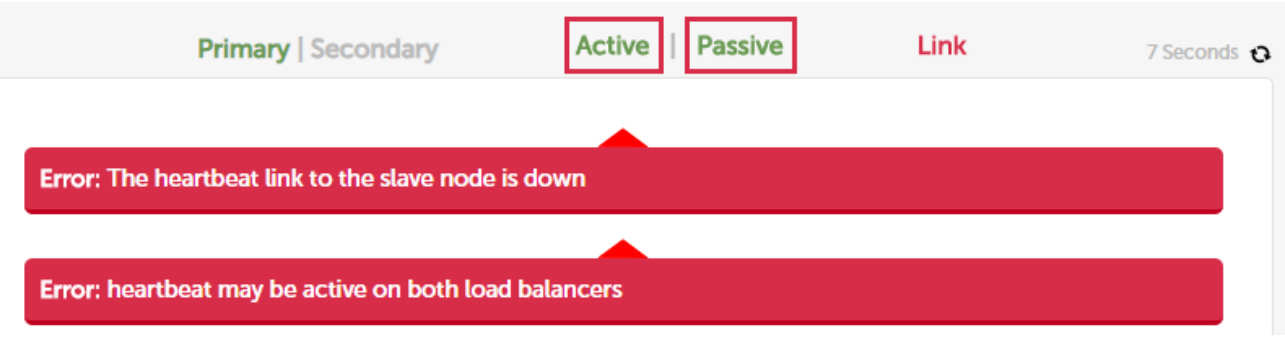


Other states:

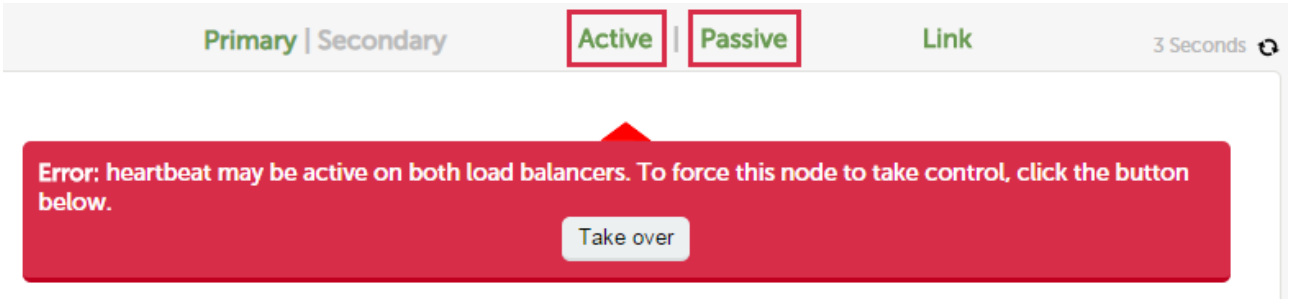
Primary | Secondary	Active | Passive	Link	this is a Primary unit, it's active, no Secondary unit has been defined.
Primary | Secondary	Active | Passive	Link	this is a Primary unit, it's active, a Secondary has been defined but the link to the Secondary is down. ***Action:*** *check & verify the heartbeat configuration & if required restart heartbeat on both units.*
Primary | Secondary	Active | Passive	Link	this is a Secondary unit, 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 unit, a Secondary unit has been defined, 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 units. ***Action:*** *check & verify the heartbeat configuration, check the serial cable (if applicable), check heartbeat logs & if required restart heartbeat on both units.*

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 units 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 units:



When heartbeat communication is re-established, heartbeat will automatically attempt to resolve the split brain and ensure that only one of the units is active. If heartbeat fails to do this automatically, the system status will show as follows on both units:

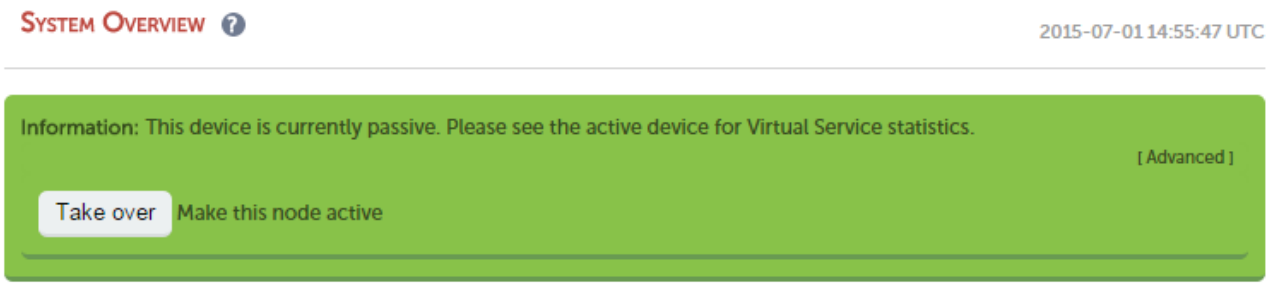


The **Take over** button can then be used on either Primary or Secondary to attempt to force that unit to become active.

Forcing Primary/Secondary Failover & Failback

To force the Secondary to become active & the Primary to become passive:

Either use the **Take over** button in the Secondary's system overview:



Note	Click the [**Advanced**] link to show this button.

Or run the following command on the Secondary:

```
/usr/local/sbin/hb_takeover.php all
```

To force the Primary to become active & the Secondary to become passive:

Either use the **Take over** button on the Primary as explained above or run the following command on the Primary:

```
/usr/local/sbin/hb_takeover.php all
```

Note	These commands 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 option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, enable 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 units 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

a) On the Primary unit verify that the system status appears as follows:

b) On the Secondary unit verify that the system status appears as follows:

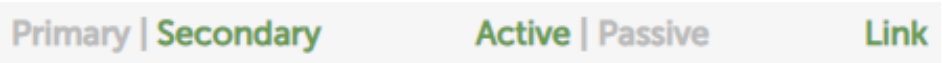
STEP 2 - Verify Replication

a) Verify that the load balanced services have been replicated to the Secondary unit, 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)

a) On the Secondary unit, click the [**Advanced**] option in the green information box, then click the **Take Over** button

b) Verify that the Secondary's status changes to *Active:*



c) Verify that the Primary's status changes to *Passive:*

d) 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 unit and brought down on the Primary

e.g. the partial screen shot below from the View Network Configuration screen on the Secondary unit shows the

status of eth0:

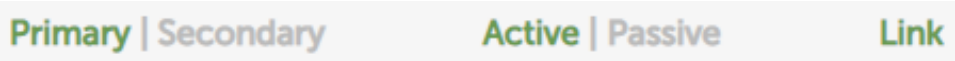
```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:0c:29:92:18:4f brd ff:ff:ff:ff:ff:ff
    inet 192.168.111.223/18 brd 192.168.127.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet 192.168.111.72/18 brd 192.168.127.255 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 Fallback to the Primary (using the Take over button)

a) On the Primary unit, click the [**Advanced**] option in the green information box, then click the **Take Over** button

b) Verify that the Primary's status has changed to *Active:*



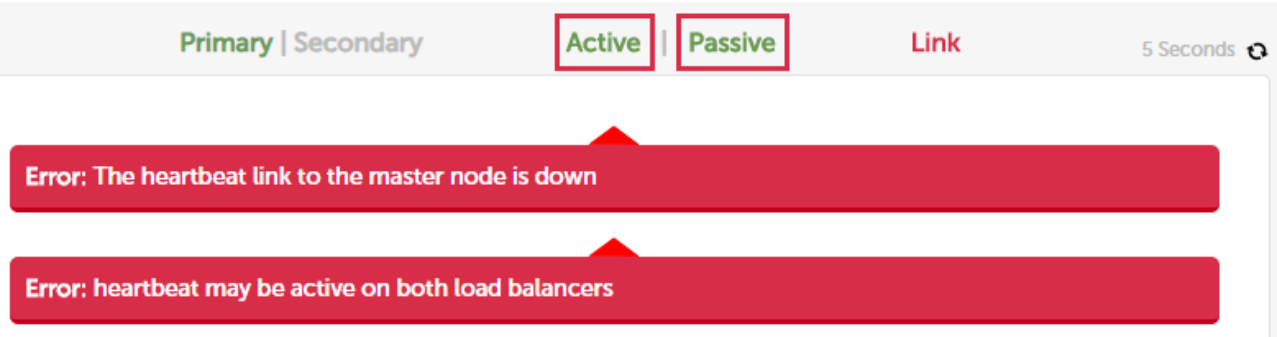
c) Verify that the Secondary's status has changed to *Passive:*

d) 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 unit 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 cable from Primary)

a) Remove the network cable and serial cable (if applicable) from the Primary

b) 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 go active.

c) 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 cable to Primary)

a) Reconnect the cables to the Primary

b) Verify that the Primary's status is set to *Active:*

c) Verify that the Secondary has changed to *Passive:*

d) 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 unit and brought down on the Secondary

Note

If the power cable on the Primary had been removed rather than disconnecting the network cable and serial cable (if applicable), once the Primary is brought back up the Secondary would remain active and the Primary would come back up in a passive state. The **Take over** button on the Primary would then need to be used to force the Primary to become active.

Chapter 10 - Application Specific Settings

FTP

FTP is a multi-port service in both active and passive modes:

active 20,21
passive 21,high_port

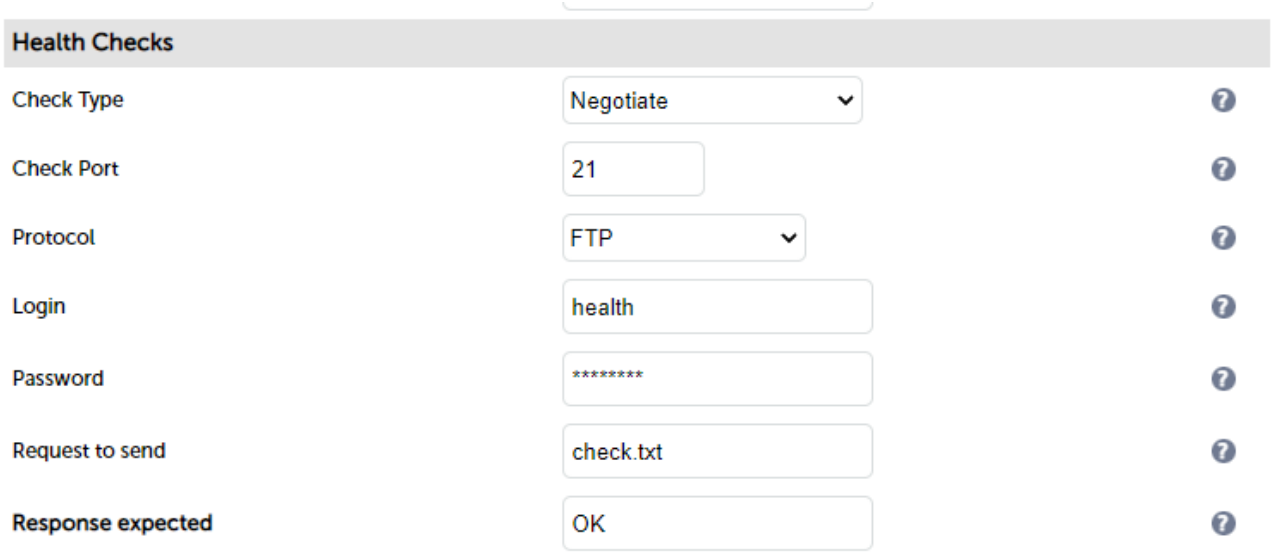
Layer 4 Virtual Services for FTP

When configuring a Virtual Service at layer 4 for FTP, simply setup a layer 4 VIP in the normal way and set the Virtual Service/Real Server port field to port 21. Where Firewall Marks are required to handle other FTP ports, these will be configured automatically. This applies to both active and passive mode. In NAT mode, the ip_vs_ftp module is used to ensure that the client connects back via the load balancer rather than attempting to connect directly to the Real Server.

Note	For VIPs configured in this way, the checkport is automatically set to port 21.

FTP Layer 4 Negotiate Health Check

You can modify the layer 4 Virtual Service so that rather than doing a simple socket connect check, it will attempt to log into the FTP server and read a file for a specific response:



Key Points:

- Change the *Check Type* to **Negotiate**
- Ensure the *Check Port* is set to **21**
- Make sure the *Negotiate Check Service* is set to **FTP**
- Specify a suitable *login* and *password* for the FTP server
- Specify the file to check using the *Request to Send* field (defaults to the root directory)
- The file is parsed for the *Response Expected* that you specify

FTP Recommended Persistence Settings

When using multiple FTP servers in a cluster you should be aware of the effects of a client switching to a different server. For sites that are download only, you generally don't need any special settings on the load balancer as the connection will usually stay on the same server for the length of the connection. You may wish to set persistence to a higher value than the default value of 5 minutes.

If you are using the FTP servers for upload it is recommended to use a single FTP server for uploads and then replicate the data to the read only cluster for downloads (or use a clustered file system). For upload it is especially important to use persistence.

Automatically resuming a broken download is no problem even if you switch servers in a cluster on re-connect. This is because the FTP resume functionality is client based and does not need any server session information.

Layer 7 Virtual Services for FTP

Active Mode

In active mode, the FTP server connects back to the client, so it must be aware of the clients 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 Server (VIP) in one subnet, the RIPs (i.e. the FTP servers) in another. 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 must be defined in the firewall script for each FTP server. 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
```

(one rule must be added 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*.

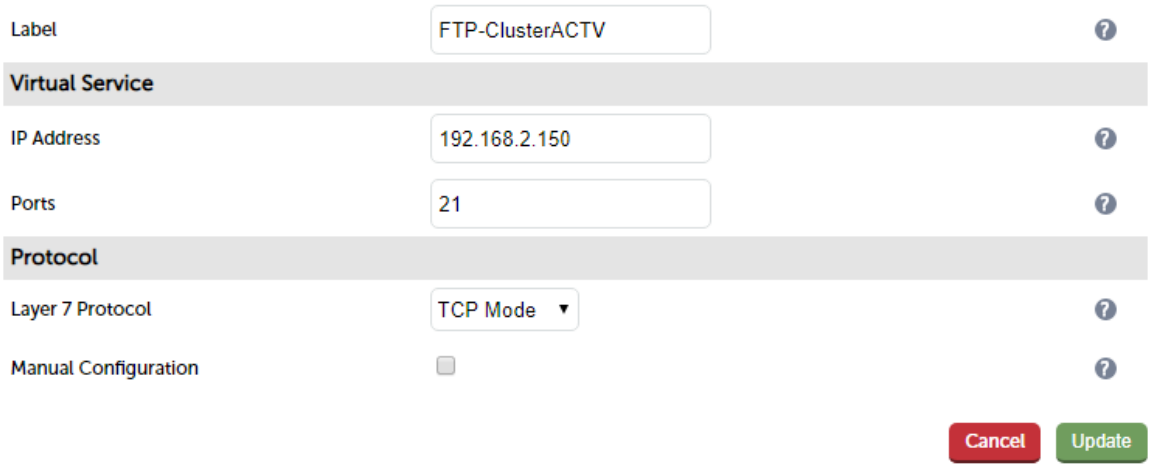
Active Mode - Key Points:

- Use separate subnets for the VIP & RIPs
- Enable TProxy for the layer 7 VIP
- Set the default gateway on the FTP servers to be an IP on the load balancer (ideally a floating IP to permit failover to the Secondary unit)
- Setup a layer 7 VIP listening on port 21 & configure the RIPs also 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

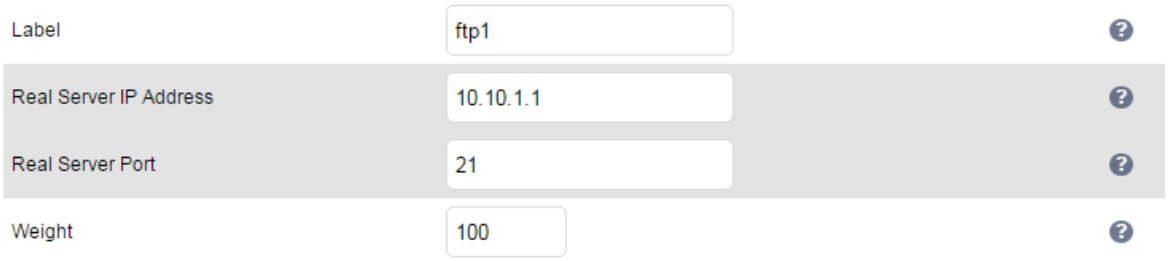
- Add the SNAT firewall rules for each FTP server

Windows 2008 R2 Example

1. Create a L7 VIP with the following settings, changing the name and IP address as required:



2. Click **Update** to create the VIP.
3. Click **Modify** next to the newly created VIP.
4. Scroll down to the section *Other* and click [**Advanced**].
5. Enable (check) the *Timeout* checkbox then set both *Client Timeout* and *Server Timeout* to **5m**.
6. Define the FTP servers as RIPs for the VIP just created as illustrated below (these must be on a different subnet to the VIP to enable TProxy to work correctly):



7. Enable TProxy using the WebUI menu option: *Cluster Configuration > Layer 7 - Advanced Configuration*.
8. Now restart HAProxy using the WebUI menu option: *Maintenance > Restart Services*.
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 192.168.2.180
iptables -t nat -A POSTROUTING -p tcp -s 10.10.1.2 -j SNAT --to-source 192.168.2.180
```

10. Configure the default gateway on each FTP server to be the load balancer. Ideally 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 (192.168.2.180) 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 Limiting Passive FTP Ports.

> **Note** This method configures HAProxy to listen on port 21 (control channel) and all passive ports (data channel).

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 passive range selected, e.g. 50000-50100
- Configure the RIPs without specifying a port
- Ensure the Layer 7 Protocol is set to 'TCP Mode'
- If transparency is required (for passive mode this is optional), enable TProxy. 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*.

> **Note** If TProxy is enabled, certain topology requirments must be met. Also the default gateway on each FTP server must also be set to be an IP on the load balancer - preferably a floating IP which then allows failover to the Secondary unit. For More information on using TProxy please refer to Transparency at Layer 7.

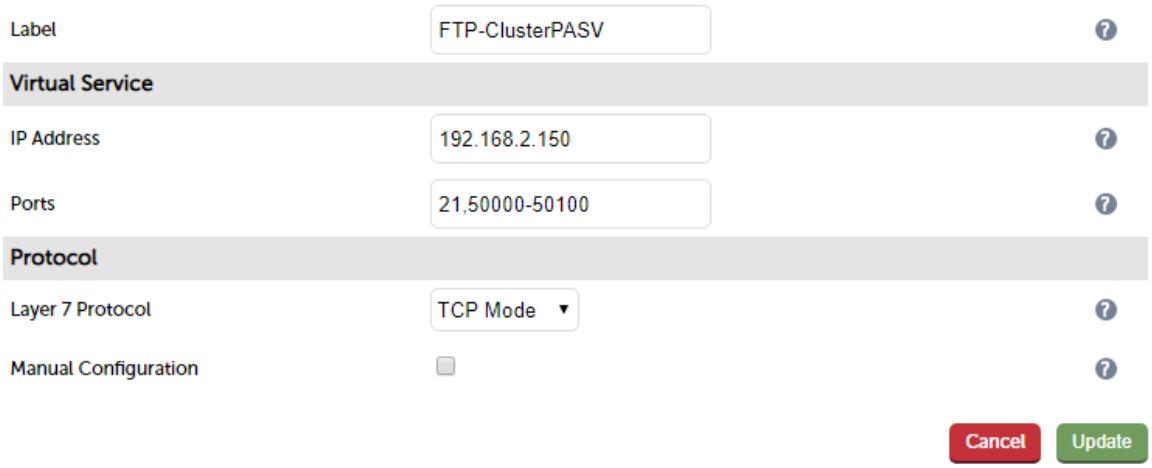
- Set the Client Timeout & Real Server Timeout to 5m (i.e. 5 minutes)
- Set the Persistence Mode to Source IP
- The Persistence Timeout can be left set to 30 (i.e. 30 minutes)
- To ensure the correct address is passed back to the client, on each FTP server specify the external address to be the VIP address.

e.g.

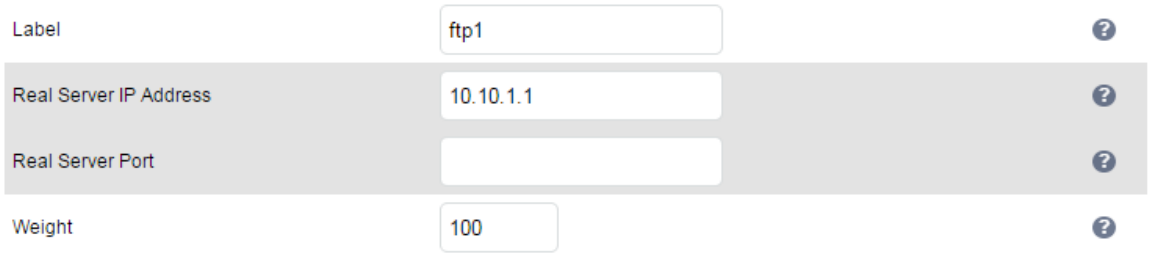
- for Windows 2008 R2 use the **External IP address of Firewall** field
- for Linux vsftpd use the directive: **pasv_address=xxx.xxx.xxx.xxx**
- for Linux ProFTPd use the directive: **MasqueradeAddress=xxx.xxx.xxx.xxx**

Windows 2008 R2 Example

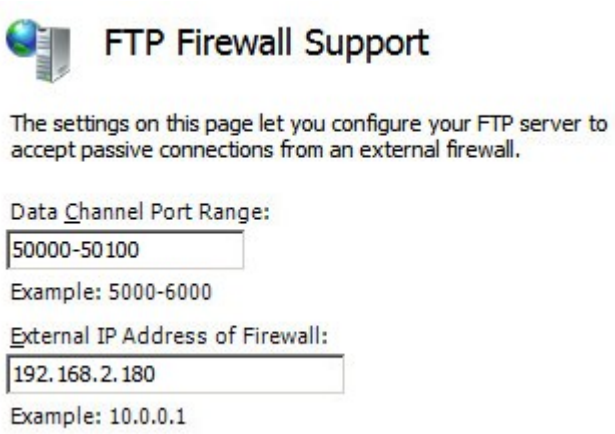
1. Create a L7 VIP with the following settings changing the name, IP address & passive port range as required:



2. Configure the VIP to listen on both the control port (21) and passive 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 section *Other* and click [**Advanced**].
6. Enable (check) the *Timeout* checkbox then set both *Client Timeout* and *Server Timeout* to **5m**.
7. Define the FTP servers as RIPs for the VIP just created leaving the port field blanks as illustrated below:



8. Now restart HAProxy using the WebUI menu option: *Maintenance > Restart Services*.
9. On each FTP server using IIS Manager define the same passive port range and set the external IP address to be the Virtual Server (VIP) address as shown in the example below:



Note	The external IP address must be set to be the VIP address, this ensure that this IP address is passed back to the client to use for the subsequent inbound connection.

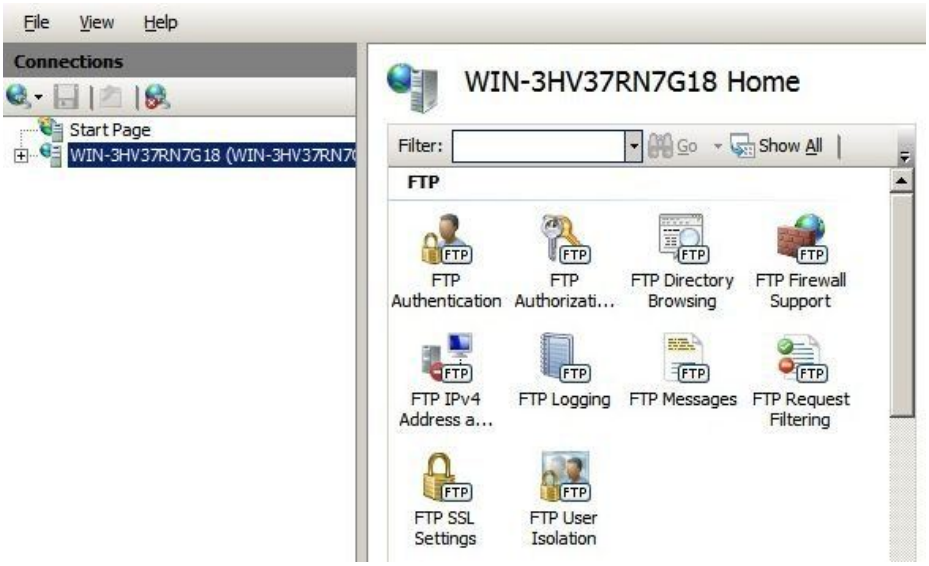
10. If TProxy is enabled, make sure the gateway of each FTP server is set to be an IP on the load balancer (preferably a floating IP to allow failover to the Secondary unit).
11. Now restart both IIS **and** the Microsoft FTP Service on each FTP server.
12. Passive FTP clients should now be able to connect to the VIP address (192.168.2.180) and view the directory listing successfully.

Limiting Passive FTP Ports

Limiting passive ports allows your firewall to be more tightly locked down. The following sections show how this is achieved for a range of Operating Systems/FTP servers.

For Windows 2008 R2 & Later

Open the IIS Management console, highlight the server node, then double-click the FTP Firewall Support icon.



The following screen will be displayed:

Specify a suitable range, in the example above this is 50000-50100

Important	Make sure you restart IIS **and** the Microsoft FTP Service to apply these settings.

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
```

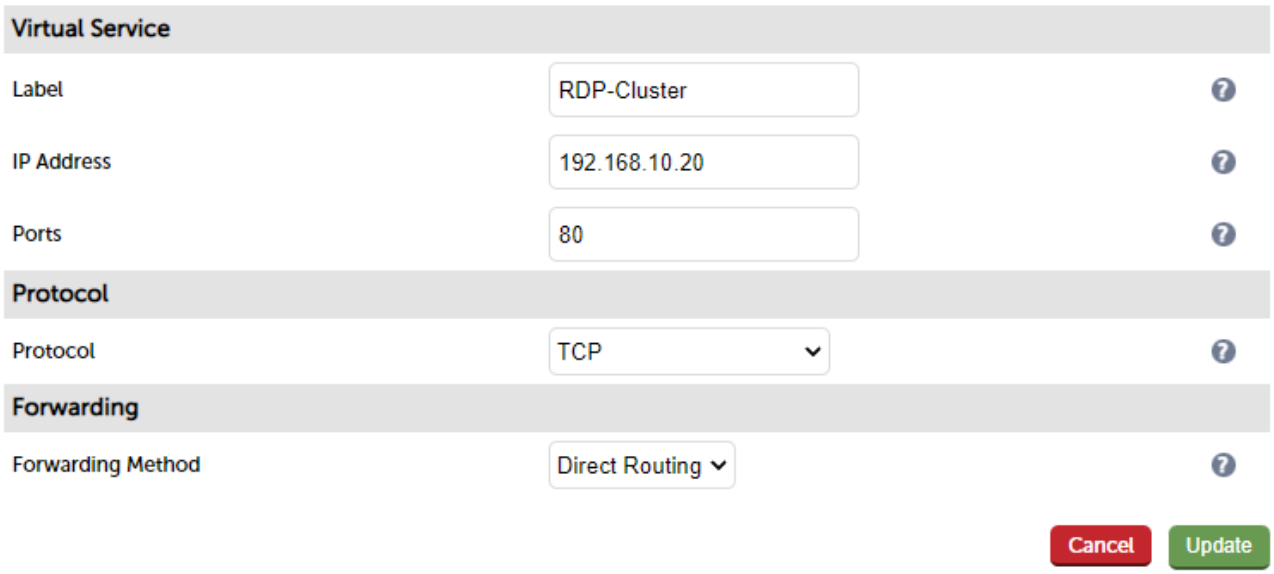
- For **pureftpd**, the following startup switch can be used:

```
-p --passiveportrange <min port:max port>
```

Terminal Services/Remote Desktop Services

Layer 4 - IP Persistence

RDP is a TCP based service usually on port 3389. Clients will need to be sent to the same server to allow re-connection to existing sessions. The persistence timeout setting can be changed to suit your requirements. A typical setting to use is 7200 (i.e. 7200s = 2 hours). This means that when a client reconnects within this time, they will be sent to the same Terminal Server/Remote Desktop Server. If a client is idle for more than 2 hours, then the load balancer will treat the next connection as a new connection and possibly take them to a different server.

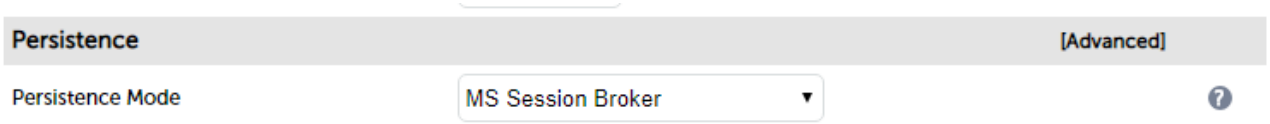


The persistence settings are accessed when the VIP is modified:

Layer 7 - Microsoft Connection Broker/Session Directory

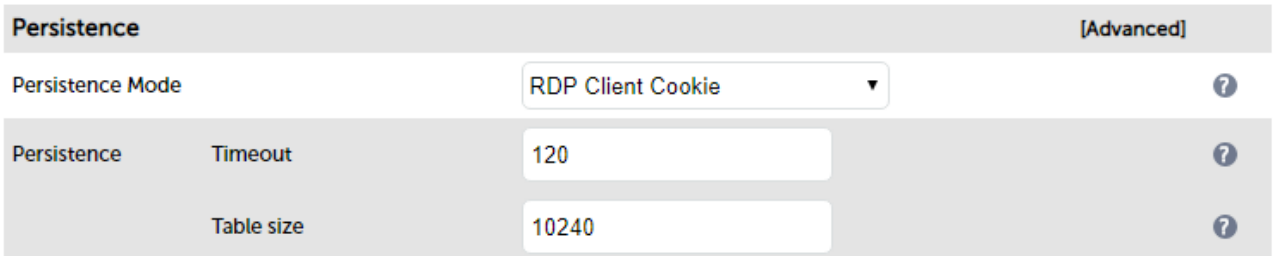
It's possible to configure the load balancer to interact with Session Directory/Connection Broker by enabling Routing Token Redirection mode. This mode allows the re-connection of disconnected sessions by utilizing a routing token to enable the load balancer to re-connect the client to the correct server. To use this kind of

persistence, create a layer 7 VIP and set the persistence mode to MS Session Broker as shown below:



Layer 7 - RDP Cookies

The appliance also supports persistence based on RDP cookies. This method utilizes the cookie sent from the client in the initial Connection Request PDU (mstshash). This cookie is created when the username is entered at the first client login prompt (mstsc.exe). Note that if the username is not entered here, the cookie is not created. An associated persistence entry is also created in a stick table on the load balancer for each connection. If the cookie is not found, it will fallback to source IP persistence. To use this kind of persistence, create a layer 7 VIP and set the persistence mode to RDP Client Cookie as shown below:



The persistence timeout can be set as required, but as per the previous example 2 hours (120m) has been configured as shown in the example above.

Initial connections are distributed to the Real Servers based on the balance mode selected. Re-connecting clients utilize the stick table to return the client to the same server first connected to. This enables clients to reconnect to their disconnected sessions.

Note

For much more information, please refer to the Remote Desktop Services Deployment Guide and the Terminal Services Deployment Guide.

Other Applications

The appliance is able to support virtually any TCP or UDP based protocol which enables most applications to be load balanced. A full list of deployment guides currently available can be found here.

Note

Please don't hesitate to contact support@loadbalancer.org for advice on load balancing your application if it's not listed.

Chapter 11 - Configuration Examples

Introduction

This section presents 4 example configurations that illustrate how the appliance is configured.

Initial Network Settings

For more information on configuring initial network settings please refer to Configuring Initial Network Settings. For more information on how to access the WebUI please refer to Accessing the WebUI.

1 - One-Arm DR Mode (Single Appliance)

This DR (Direct Return) mode example has one Virtual Service (VIP) with two Real Servers (RIPs). It's a straight forward deployment mode that can be used in many situations. It also offers the highest performance because return traffic passes directly from the Real Servers to the client rather than passing back via the load balancer.

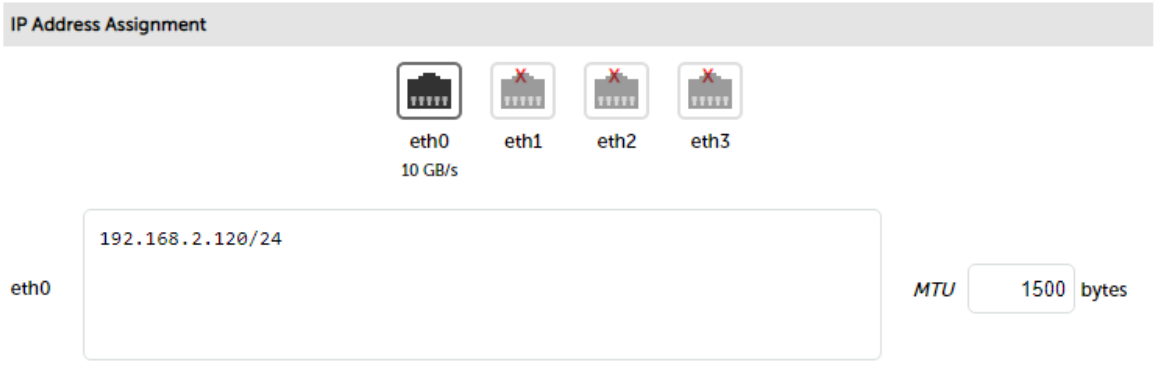
Configuration Overview

- **Configure Network Settings** - a single Interface is needed, eth0 is normally used
- **Define the Virtual Service (VIP)** - all Real (backend) Servers are accessed via this IP address
- **Define the Real Servers (RIPs)** - define the Real Servers that make up the cluster
- **Implement the required changes to the Real Servers** - for DR mode, the ***ARP Problem*** must be solved

Network Settings

Configure the various network settings as outlined below:

1. Using the WebUI, navigate to: *Local Configuration > Network Interface Configuration*.
2. Scroll down to the *IP Address Assignment* section.



3. Specify the IP address & subnet mask for eth0 (normally eth0 is used for single-arm configurations although this is not mandatory), e.g. **192.168.2.120/24**.
4. Click **Configure Interfaces**.
5. Using the WebUI, navigate to: *Local Configuration > Hostname & DNS*.
6. Specify the DNS server(s).

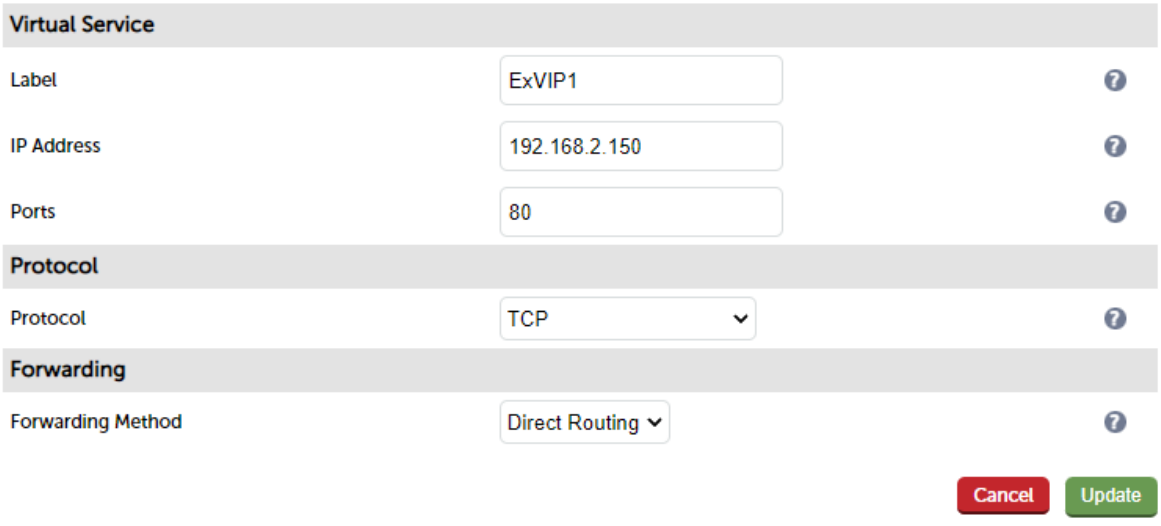
7. Click **Update**.
8. Using the WebUI, navigate to: *Local Configuration > Routing*.

9. Specify the Default Gateway.
10. Click **Configure Routing**.

Virtual Service (VIP)

Next, configure the Virtual Service. This is the IP address that is presented to clients. Any packet arriving at the load balancer with that IP address and port number will be forwarded to the the Real Servers associated with the Virtual Service. This example is for Web traffic on TCP port 80.

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



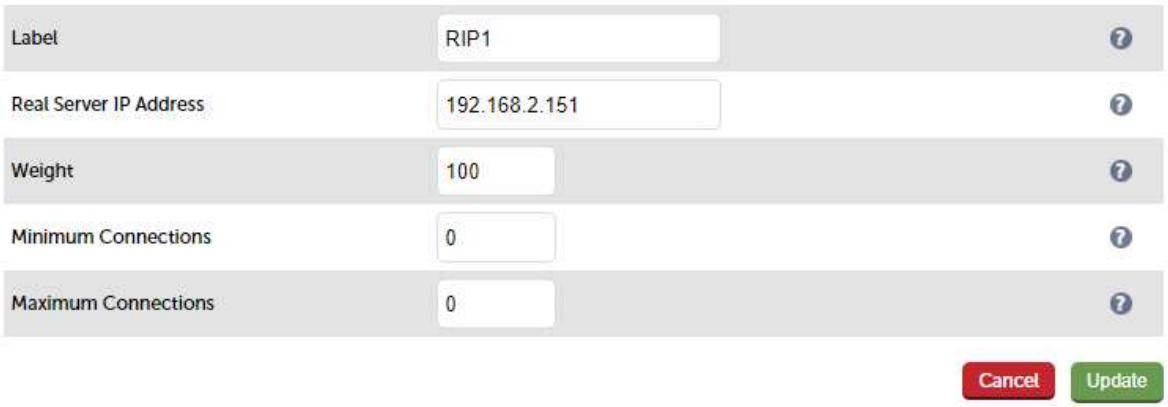
2. Enter a suitable Label (name) for the VIP, e.g. **ExVIP1**.
3. Enter a valid IP address, e.g. **192.168.2.150**.
4. Enter a valid port, e.g. **80**.
5. Select the required Protocol, .e.g. **TCP**.
6. Ensure that the *Forwarding Method* is set to **Direct Routing**.

7. Click **Update**.

Real Servers (RIPs)

Each Virtual Service requires a cluster of Real Servers (backend servers) to forward the traffic to.

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



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. The port used will be the same as that configured for the VIP.

4. The weight defaults to 100 making the Real Server active immediately.
5. Leave *Minimum Connections* & *Maximum Connections* set to 0 which means unrestricted.
6. Click **Update**.
7. Repeat for the other Real Server.

Physical Real Server Changes - Solve the ARP Problem

For DR mode, the ***ARP Problem*** must be solved on each Real Server:

- Each Real Server must be configured to respond to its own IP address **and** the VIP address
- Each Real Server must be configured so that it only responds to ARP requests for its own IP address, it should **not** respond to ARP requests for the VIP address - only the load balancer must respond to these requests

> Note
>
> Failure to correctly configure the Real Servers to handle the ***ARP Problem*** is the most common problem in DR configurations. For more information on the ***ARP Problem*** and the solution for various OSs, please refer to The ARP Problem.

Basic Testing & Verification

Once configured, a few quick checks can be performed to verify the setup:

1. Using *System Overview* 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. 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.

2 - One-Arm Layer 4 SNAT Mode (Single Appliance)

This layer 4 SNAT mode example has one Virtual Service (VIP) with two Real Servers (RIPs). 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 by default, i.e. the Real Servers will see the source IP address of the load balancer.

Note	In this mode, no changes are required to the Real Servers.

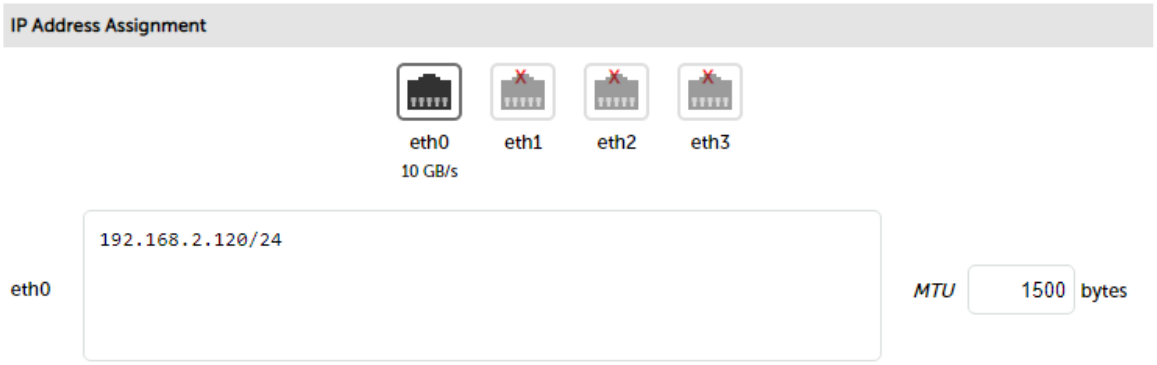
Configuration Overview

- **Configure Network Settings** - a single Interface is needed, eth0 is normally used
- **Define the Virtual Service (VIP)** - all Real (backend) Servers are accessed via this IP address
- **Define the Real Servers (RIPs)** - define the Real Servers that make up the cluster

Network Settings

Configure the various network settings as outlined below:

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



2. Specify the IP address & subnet mask for eth0 (normally eth0 is used for single-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 > Hostname & DNS*.
5. Specify the DNS server(s).

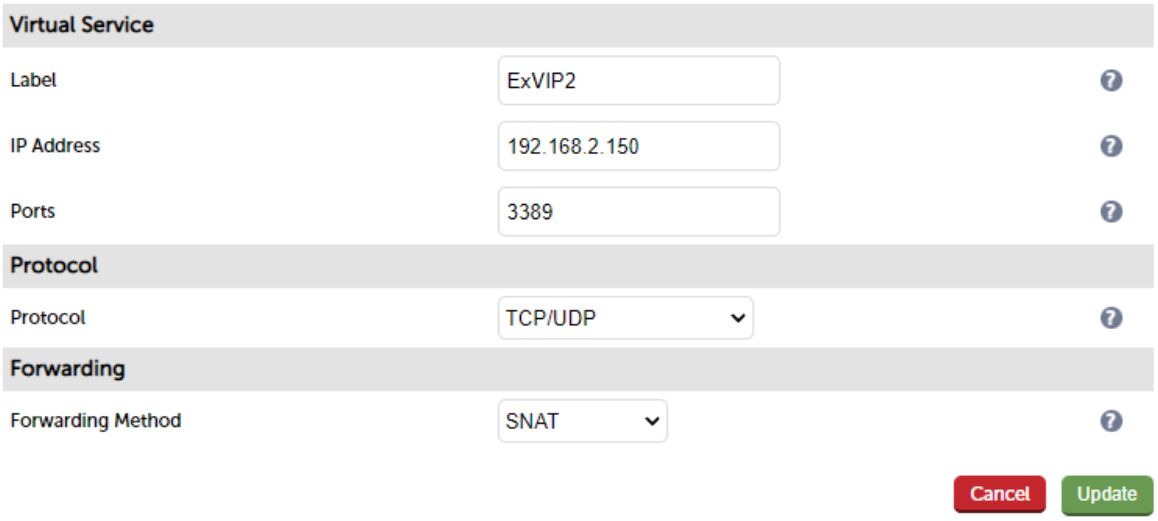
6. Click **Update**.
7. Using the WebUI, navigate to: *Local Configuration > Routing*.

8. Specify the Default Gateway.
9. Click **Configure Routing**.

Virtual Service (VIP)

Next, configure the Virtual Service. This is the IP address that is presented to clients. Any packet arriving at the load balancer with that IP address and port number will be forwarded to the the Real Servers associated with the Virtual Service. This example is for RDP traffic on TCP/UDP port 3389.

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



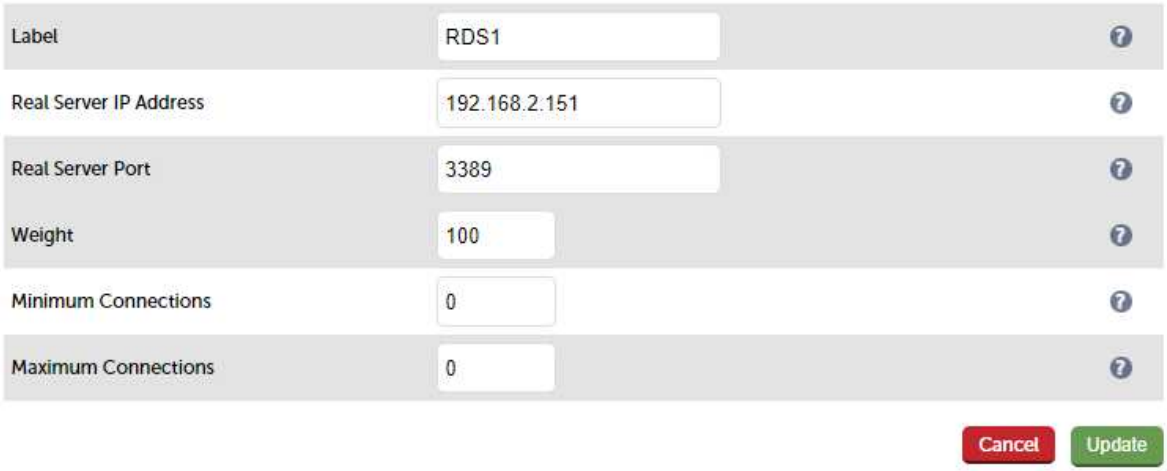
2. Enter a suitable Label (name) for the VIP, e.g. **ExVIP2**.
3. Enter a valid IP address, e.g. **192.168.2.150**.
4. Enter the required port, e.g. **3389**.
5. Select the required *Protocol* , in this example **TCP/UDP** (UDP support was added in RDP v8.0).
6. Ensure that *Forwarding Method* is set to **SNAT**.

7. Click **Update**.

Real Servers (RIPs)

Each Virtual Service requires a cluster of Real Servers (backend servers) to forward the traffic to.

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

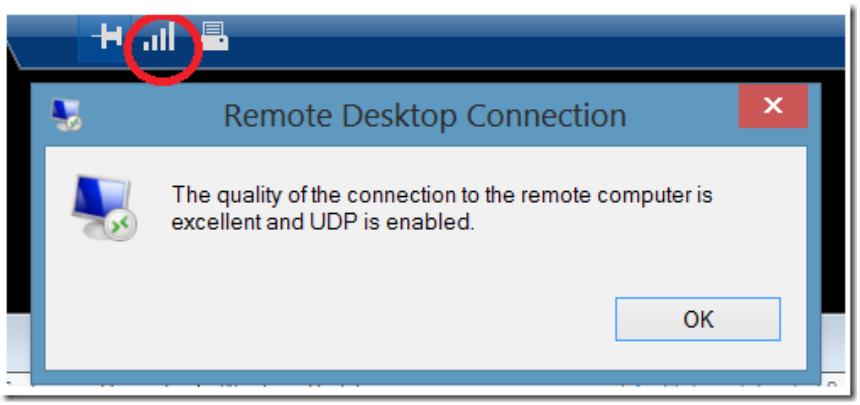


2. Enter a suitable Label (name) for the RIP, e.g. **RDS1**.
3. Enter a valid IP address, e.g. **192.168.2.151**.
4. Enter the required port, e.g. **3389**.
5. The weight defaults to 100 making the Real Server active immediately.
6. Leave *Minimum Connections* & *Maximum Connections* set to 0 which means unrestricted.
7. Click **Update**.
8. Repeat for the other Real Server.

Basic Testing & Verification

Once configured, a few quick checks can be performed to verify the setup:

1. Using *System Overview* check that the VIP & RIPs are shown as active (green).
2. Using Windows RDP client (mstsc.exe) to connect to the VIP address, i.e. **192.168.2.150** to verify that you can start an RDP session.
3. Verify that the RDP session supports TCP & UDP by clicking the connection info button on the RDS Connection Bar:



3 - Two-Arm NAT Mode (Clustered Pair)

This example shows how to configure two appliances as a clustered pair using layer 4 NAT mode.

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.

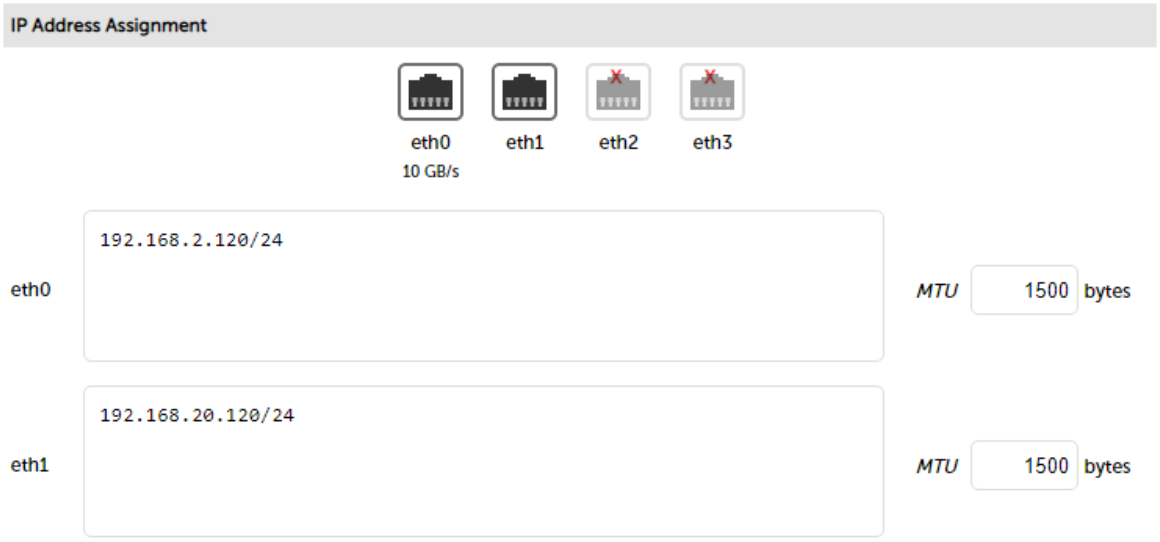
Note	When using two-arm NAT mode all Real Servers should be in the same subnet as the internal interface of the load balancer and the default gateway on each Real Server must be set to be an IP on the load balancer, for a clustered pair this should be a floating IP to allow failover.

Configuration Overview

- **Configure the Primary's Network Settings** - two Interfaces are needed, this can be either two physical interfaces such as eth0 and eth1, or one physical interface and a secondary interface/alias
- **Configure the Secondary's Network Settings** - two Interfaces are needed, this can be either two physical interfaces such as eth0 and eth1, or one physical interface and a secondary interface/alias
- **On the Primary, Define the Virtual Service (VIP)** - all Real Servers are accessed via this IP address
- **On the Primary, Define the Real Servers (RIPs)** - define the Real Servers that make up the cluster
- **Implement the required changes to the Real Servers** - in NAT mode, the Real Servers default gateway must be set to be the load balancer
- **Create the HA Clustered Pair** - pair the Primary & Secondary to synchronize the appliances
- **Verify Heartbeat Settings** - check that the default heartbeat settings are appropriate

Primary Unit - Network Settings

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



2. Specify the IP address & mask for eth0 - normally eth0 is configured as the *internal* interface although this is not mandatory, e.g. **192.168.2.120/24**.
3. Specify the IP address & mask for eth1 - normally eth1 is configured as the *external* interface although this is not mandatory, e.g. **192.168.20.120/24**.

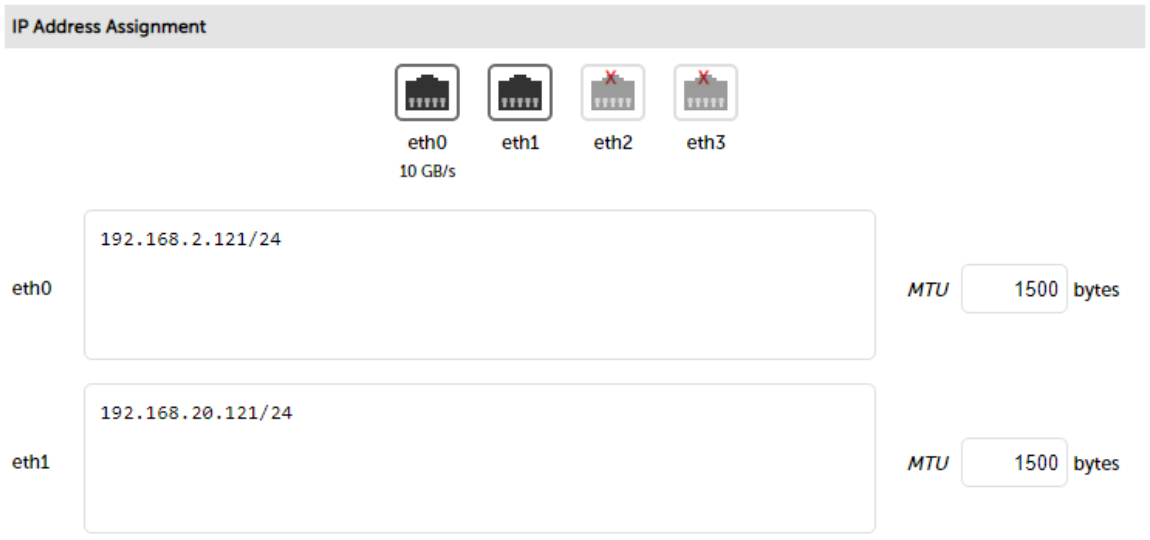
> Note
>
> For a VA make sure that the virtual NIC associated with eth1 is connected to the virtual switch, by default only the first NIC is connected.

4. Click **Configure Interfaces**.

Secondary Unit - Network Settings

Configure the various network settings as outlined below:

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



2. Specify the IP address & mask for eth0 - normally eth0 is configured as the *internal* interface although this is not mandatory, e.g. **192.168.2.121/24**.

3. Specify the IP address & mask for eth1 - normally eth1 is configured as the *external* interface although this is not mandatory, e.g. **192.168.20.121/24**.

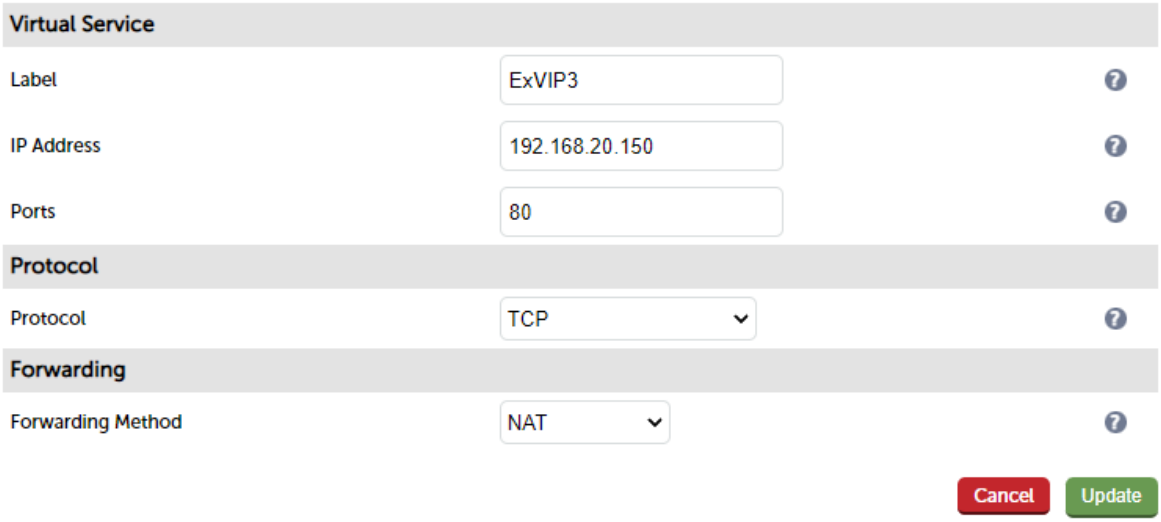
> Note
>
> For a VA make sure that the virtual NIC associated with eth1 is connected to the virtual switch, by default only the first NIC is connected.

4. Click **Configure Interfaces**.

Virtual Service (VIP)

Next, configure the Virtual Service. This is the IP address that is presented to clients. Any packet arriving at the load balancer with that IP address and port number will be forwarded to the the Real Servers associated with the Virtual Service. This should be done on the Primary appliance.

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



2. Enter a suitable label (name) for the VIP, e.g. **ExVIP3**.
3. Enter a valid IP address, e.g. **192.168.20.150**.
4. Enter a valid port, e.g. **80**.
5. Ensure that *Forwarding Method* is set to **NAT**.
6. Click **Update**.

Real Servers (RIPs)

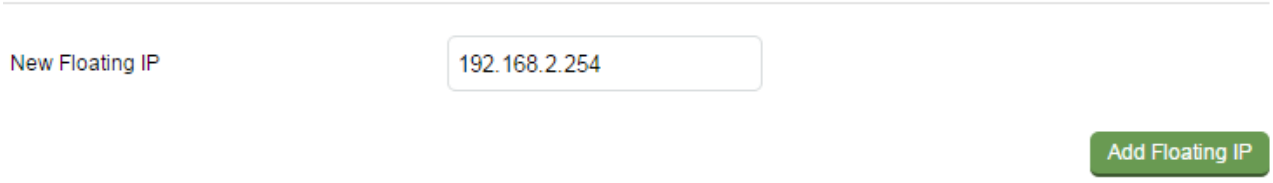
Each Virtual Service requires a cluster of Real Servers (backend servers) to forward the traffic to. This should be done on the Primary appliance.

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 4 - Real Servers* and click **Add a new Real Server**.

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. *Weight* defaults to 100 making the Real Server active immediately.
6. Leave *Minimum Connections* & *Maximum Connections* set to 0 which means unrestricted.
7. Click **Update**.
8. Repeat for the other Real Server.

Physical Real Server Changes - Set the Default Gateway

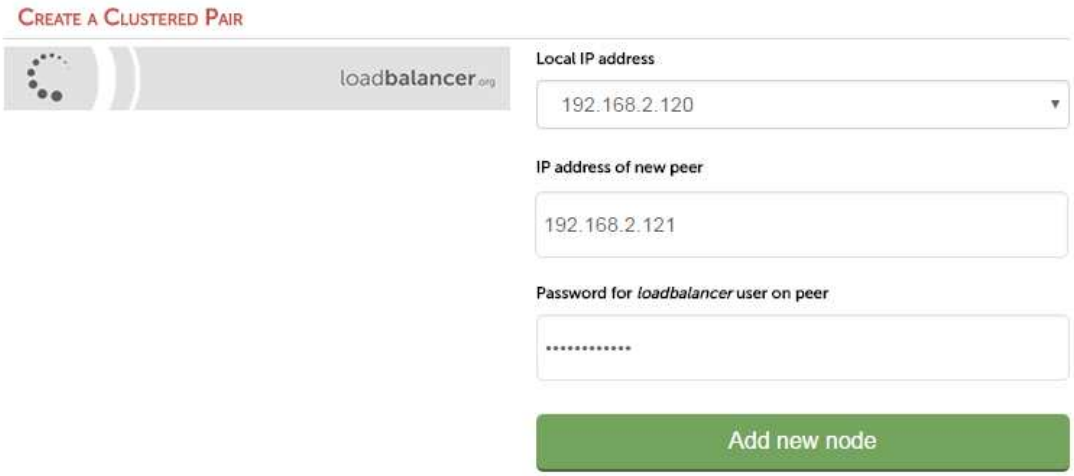
When using NAT mode, each Real Server's default gateway must be changed to be the load balancer. For a clustered pair, you must define an additional floating IP for this purpose. Then, if failover occurs, the same IP will also be brought up on the Secondary. To add a floating IP to use as the default gateway, use *Cluster Configuration > Floating IPs.*



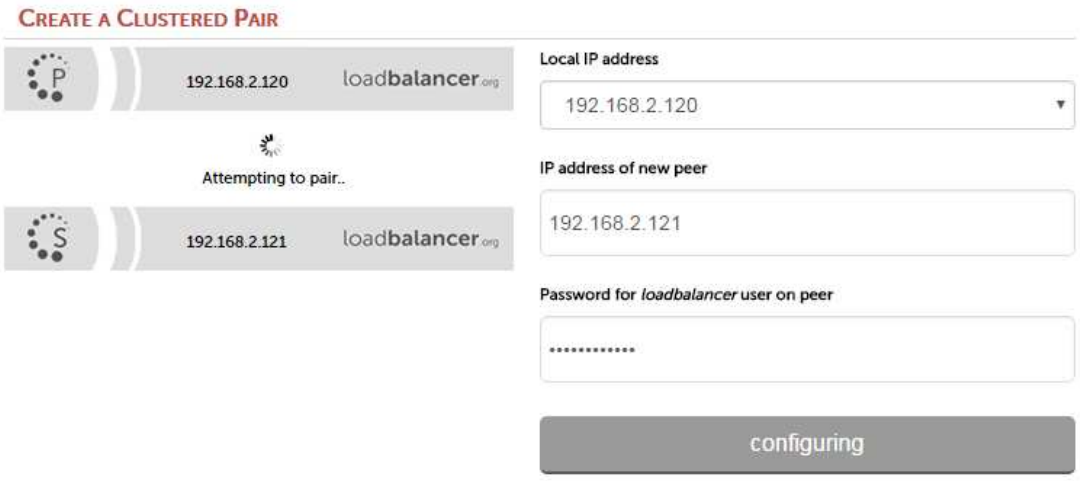
Define the IP address that you'd like to use for the default gateway, then click **Add Floating IP**. Now configure the default gateway on each Real Server to use this address.

Create the HA Clustered Pair

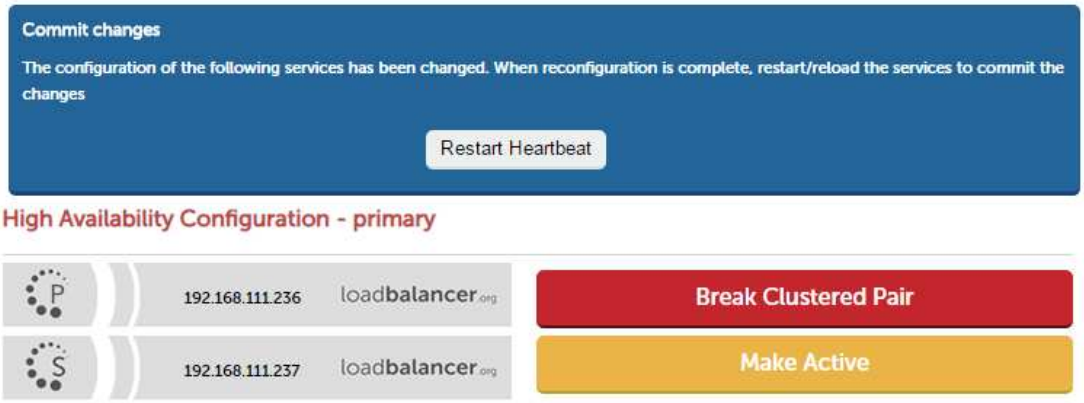
1. Using the WebUI on the Primary appliance , navigate to: *Cluster Configuration > High Availability Configuration*



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* users password (the default is 'loadbalancer') 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:



8. Once completed successfully, the following message will be displayed:

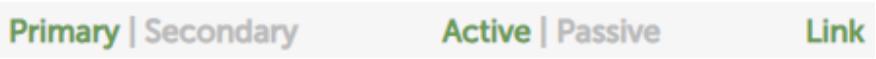


9. To finalize the configuration, click **Restart Heartbeat**.

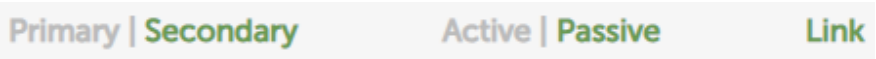
Checking the Status

A successfully configured clustered pair will display the following status:

1) On the Primary unit verify that the system status appears as follows:



2) On the Secondary unit verify that the system status appears as follows:



Note	Once the VIP has been defined, the Active text will be colored green on the Primary and Passive will be colored green on the Secondary.

Verify Heartbeat Settings

1. Using the WebUI on the Primary appliance , navigate to: *Cluster Configuration > Heartbeat Configuration*.
2. The default Heartbeat settings are normally fine for most situations. For details of all heartbeat options please refer to Configuring Heartbeat.

Verify the Secondary Configuration

To verify that the new VIP & RIP have been replicated correctly, open the WebUI on the Secondary and navigate to: *Cluster Configuration > Layer 4 - Virtual Services* and *Cluster Configuration > Layer 4 - Real Servers* and check that your configuration appears there also. For a correctly configured pair, the VIPs and RIPs are automatically replicated to the Secondary as they are defined on the Primary.

If not, double check that both units are configured correctly and that the IP address for the Secondary defined on the Primary is correct. Then on the Primary navigate to: *Maintenance > Backup & Restore* and click *Synchronize* **Configuration with peer**. This will force the VIPs & RIPs to be copied from the Primary to the Secondary, then check again.

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 - One-Arm SNAT Mode & SSL Termination (Single Appliance)

This example uses HAProxy and STunnel at layer 7. STunnel is used to terminate SSL on the load balancer. STunnel then passes unencrypted HTTP traffic to the HAProxy VIP/RIP cluster. HAProxy does not offer the raw throughput of layer 4, but is still a high performance solution that is appropriate in many situations.

Note	Pound can also be used for SSL termination, although STunnel is the preferred and default method.

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 will be 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 then new connections may get distributed to a different Real Server which may result in failure of the application.

Note	Because HAProxy is a full proxy, any server in the cluster can be on any accessible subnet including across the Internet or WAN.

Note	In this mode, no changes are required to the Real Servers.

Note	We generally recommend that SSL is terminated on the Real Servers rather than on the load balancer. This ensures that the SSL load is distributed and also ensures scalability.

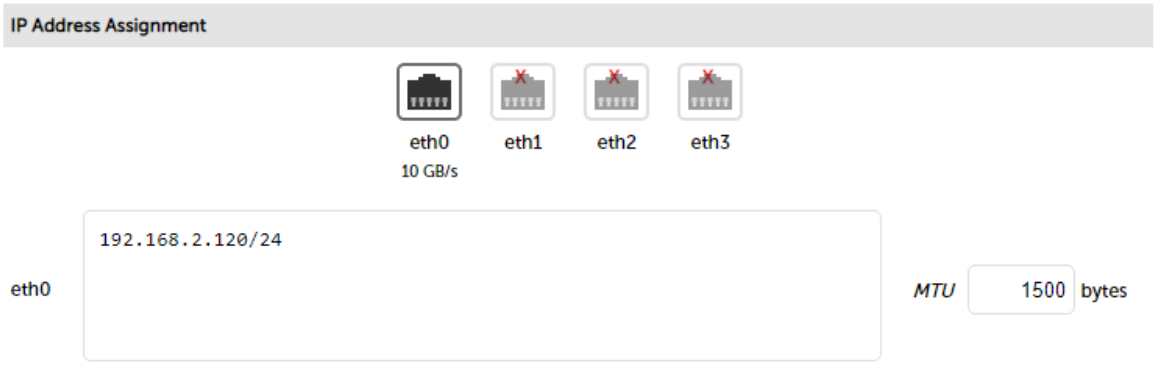
Configuration Overview

- **Configure Network Settings** - A single Interface is needed, eth0 is normally used
- **Define the Virtual Service (VIP)** - All Real Servers are accessed via this IP address
- **Define the Real Servers (RIPs)** - Define the Real Servers that make up the cluster
- **Configure SSL Termination** - Configure STunnel for SSL termination

Network Settings

Configure the various network settings as outlined below:

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



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).

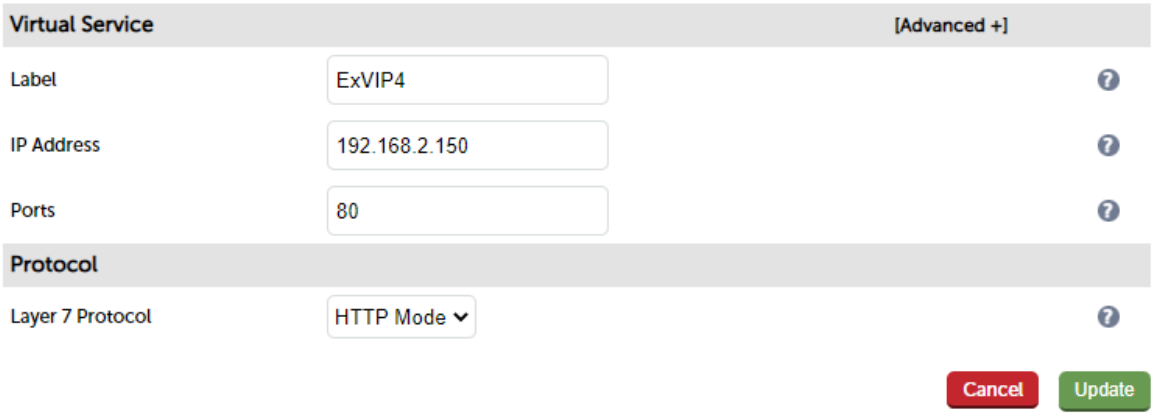
6. Click **Update**.
7. Using the WebUI, navigate to: *Local Configuration > Routing*.

8. Specify the Default Gateway, e.g. 192.168.2.254.
9. Click **Configure Routing**.

Virtual Service (VIP)

Next, configure the Virtual Service. This is the IP address that is presented to clients. Any packet arriving at the load balancer with that IP address and port number will be handled by the Real Servers associated with the Virtual Service.

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Virtual Services* and click **Add a new Virtual Service**.



2. Enter a suitable Label (name) for the VIP, e.g. **ExVIP4**.
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**.

Real Servers (RIPs)

Each Virtual Service requires a cluster of Real Servers (backend servers) to forward the traffic to.

1. Using the WebUI, navigate to: *Cluster Configuration > Layer 7 - Real Servers* and click **Add a new Real Server**.

2. Enter a suitable Label (name) for the RIP, e.g. **RIP1**.
3. Enter a valid IP address, e.g. **192.168.2.151**.

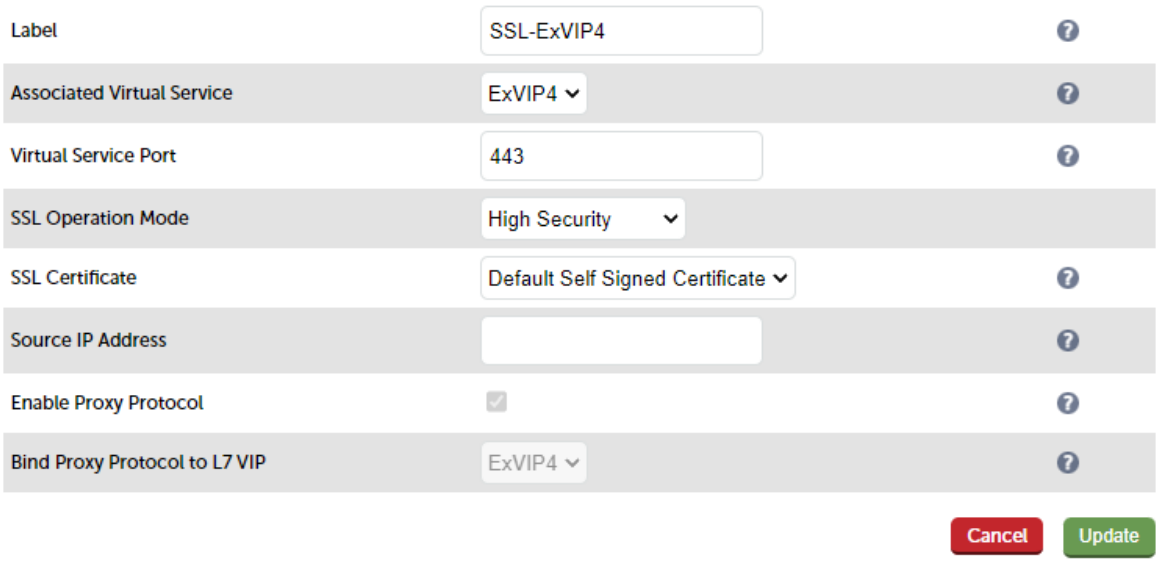
> Note
>
> In this mode it's possible to have a different port for the RIP than was configured for the VIP, in this example both are the same.

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. Repeat for the remaining Real Servers.
8. Reload HAProxy to apply the new settings using the link provided in the blue box at the top of the screen.

SSL Termination

An SSL Virtual Service is configured on port 443 using the same IP address as the Layer 7 VIP created previously. This allows a single IP address to be used for HTTP & HTTPS client connections.

1. Using the WebUI, navigate to: *Cluster Configuration > SSL Termination* and click **Add a new Virtual Service**.



2. Enter an appropriate *Label* (name) for the new Virtual Service.

> Note
>
> The label will be auto configured based on the *Associated Virtual Service* selected. This can be edited if required.

3. Set *Associated Virtual Service* to the Layer 7 VIP created earlier, e.g. **ExVIP4**.
4. Leave *Virtual Service Port* set to **443**.
5. Leave the other settings at their default values.
6. Click **Update**.
7. Reload STunnel to apply the new settings using the link provided in the blue 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 WUI option: *Cluster Configuration > SSL Certificate*. Once added, these will appear in the *SSL Certificate* drop-down 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.

Chapter 12 - Testing Load Balanced Services

Introduction

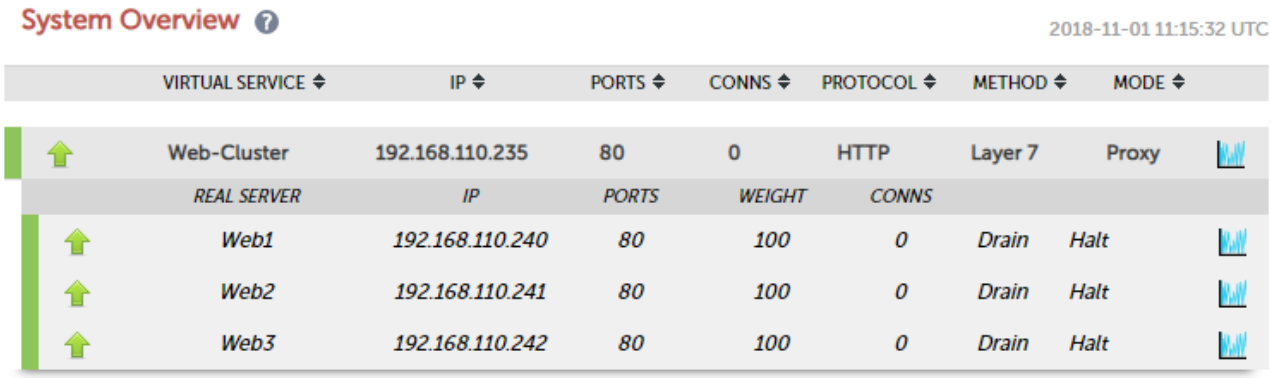
Once your load balanced services have been configured, you'll need to test and verify that everything is working as expected.

Note

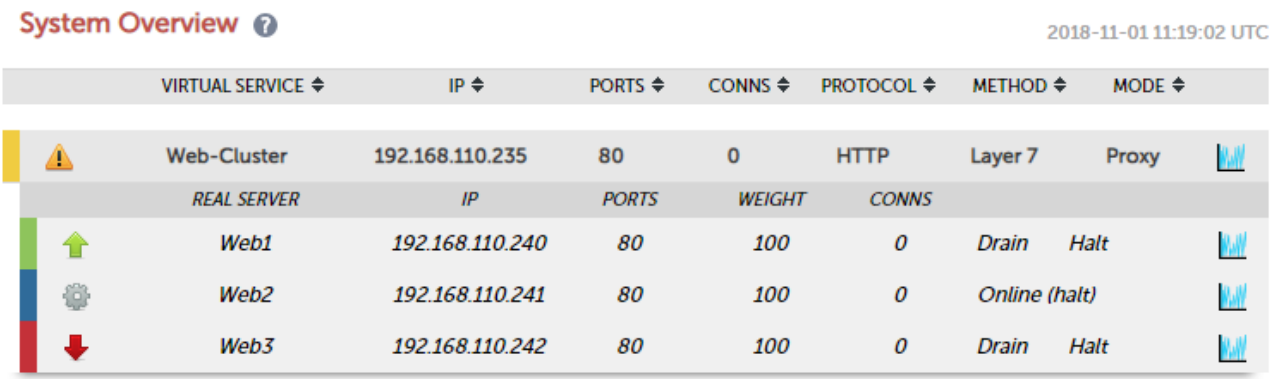
For more information on testing & verifying an HA Clustered Pair please refer to Testing & Verifying Primary/Secondary Replication & Failover.

Checking that Services are Up

A good place to start is to verify that configured services are displayed green (i.e. healthy) in the System Overview. This provides a quick way to spot any obvious issues. The first screenshot shows that the VIP *Web-Cluster* is healthy and that all associated Real Servers are up.



The second screenshot shows that the VIP is colored yellow and marked with an exclamation mark indicating that one or more of the Real Servers is not available. The colored tab and the arrow on the left show the current status of each Real Server.



In the example above:

- The Virtual Service *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)
- The Real Server *Web1* is green, this indicates that it's passing it's health check
- The Real Server *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)

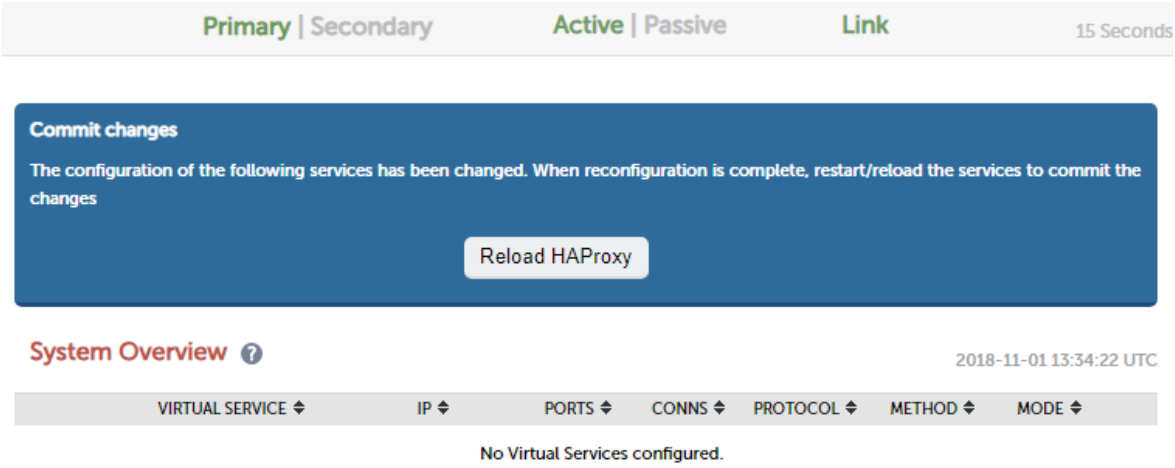
- The Real Server *Web3* is red, this indicates that it has failed it's health check

Diagnosing VIP Issues

VIP(s) Fail to appear in the System Overview

If you have configured new VIPs and these have not automatically appeared in the System Overview:

1. For layer 7 VIPs, have you restarted or reloaded HAPproxy since adding the VIP? As shown in the screen shot below, new VIPs are not displayed until a service reload or restart occurs.



2. Is the corresponding floating IP active? When a new VIP is configured, a corresponding floating IP is automatically added and brought up.
3. If all floating IPs are missing or are down, then the System status bar will show both "Active" & "Passive" colored grey, also none of the VIPs will be displayed in the System Overview.
4. Configured Floating IPs can be viewed using the WebUI option: *Cluster Configuration > Floating IPs*.
5. The actual running network configuration can be viewed using the WebUI option: *View Configuration > Network Configuration*.

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:0c:29:62:d3:56 brd ff:ff:ff:ff:ff:ff
    inet 192.168.111.230/18 brd 192.168.127.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet 192.168.110.235/18 brd 192.168.127.255 scope global secondary eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe62:d356/64 scope link
       valid_lft forever preferred_lft forever
```

The above example shows that the interface address (192.168.111.230) and the floating IP address (192.168.110.235) associated with the VIP are both up.

When all Floating IPs are missing or are down, both "Active" & "Passive" are colored grey:

VIPs & RIPs are Green but Users Still Cannot 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 depending on whether you've configured layer 4 or layer 7 VIPs.

Layer 7 VIPs

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 like RDP on port 3389 or HTTPS on port 443, then you'll need to change the *Layer 7 Protocol* drop-down to TCP.

Is TProxy enabled? If you've enabled TProxy under *Layer 7 - Advanced Configuration* to make your layer 7 VIP transparent, there are certain topology and other requirements that must be met.

Note	For more information about enabling transparency at layer 7, please refer to Transparency at Layer 7.

Layer 4 VIPs

Have you complied with the layer 4 network topology 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 sections below 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 - any packets with state SYN_RECV imply 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 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 packets with state SYN_RECV often imply 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 rather than the interface address to allow

failover & failback.

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:

1. Verify that the application / service is running on each Real Server.
2. 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?
3. 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. The 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 option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, enable the required option(s) and click **Update**.

4. 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 option: *Local Configuration > Security*. You'll need to Set *Appliance Security Mode* to **Custom**, enable the required option(s) and click **Update**.

5. Check if there is a firewall preventing access to the Real Server.

Verifying Requests are Load Balanced as Expected

As part of your testing you'll probably want to verify that requests are being equally load balanced between your Real Servers.

Creating a Simple Test Environment

For example, to test a web server based configuration, add a page to each web server's root directory e.g. test.html and put the server name on this page for easy identification during the tests.

Open a web browser on each test client and enter the URL for the VIP e.g. **http://192.168.110.10**.

Each client should see a different server name because of the load balancing algorithm in use, i.e. they are being load balanced across the cluster.

Testing Considerations

When performing your tests, consider the following points:

1. **Use multiple test clients** - always use more than one test client to ensure representative results. if a single client is used, all requests have the same source IP address. Also certain clients (e.g. web browsers) can work in different ways in relation to closing the underlying TCP connection which can give different results.
2. **Is persistence enabled** - if persistence is enabled, a particular client should be consistently load balanced to the same Real Server during a particular session (assuming the persistence timeout has not expired).
3. **What type of persistence is enabled** - if you've selected a persistence type which is not appropriate for your environment, for example if you've selected HTTP cookie persistence for a non HTTP based service, this will effectively be the same as selecting no persistence.
4. **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.

Draining & Halting Real Servers

1. Using the *System Overview* verify that when you Drain one of the Real Servers, new connections are sent to one of other Real Servers.
2. Using the *System Overview* verify that when you Halt one of the Real Servers, all connections are handled by one of the other Real Servers.

Triggering Real Server Failures

1. Remove the network cable from one of the Real Servers or stop the application service/process, wait a few seconds (for the load balancer to detect the change) and then refresh the client application on both clients. They should now both switch to the same server (since one has been removed from the load balancing list). Also check that the server is shown red (down) in the system overview.

Note

When using the default health check which is connect to port, halting some applications (e.g. IIS) can still result in a successful health check. This is because port 80 is still open and accepting new connections. In this case, a more robust negotiate check should be used to ensure that the port is open AND the application is responding.

2. Replace the network cable, wait a few seconds and then refresh the browsers again. After a few refreshes they should again show different web servers. Also check that the server is shown green (up) in the system overview.

Other Diagnostics Tools

The appliance has a number of log files and reports that may be helpful when verifying that the load balancer has been configured correctly for your environment.

Log Files

The appliance includes several log files that can be very useful when diagnosing issues. These include load balancer events, layer 4 and layer 7 specific logs and heartbeat logs. For full details of all logs, please refer to the next chapter.

Reports

The appliance includes several reports that can be very useful when diagnosing issues. These include the Layer 4 Status Report and the Layer 7 Status Report. For full details of all reports, please refer to the next chapter.

Chapter 13 - Appliance Monitoring

Appliance Log Files

All appliance logs can be accessed using the *Logs* option in the WebUI.

Note	Appliance log files can be downloaded for external analysis using the **Download Log** button that is available at the top of each log page.

Load Balancer

/var/log/lbadmin.log

The lbadmin log shows all changes made to the appliances configuration. This is very useful for tracking changes made to the configuration.

Layer 4

/var/log/ldirectord.log

The Ldirectord log shows the output from the health checking daemon. This is useful for checking the health of your Real Servers or pinning down any configuration errors. The logging here can be quite verbose but it clearly shows exactly what the health checking process is doing.

Layer 7

/var/log/haproxy.log

If activated via *Cluster Configuration > Layer 7 - Advanced Configuration*, this will show the contents of the HAProxy log. This is a very detailed log of all HAProxy transactions. It's also possible to configure HAProxy to log errors only.

SSL Termination (Pound)

/var/log/poundssl.log

If activated via *Cluster Configuration > SSL - Advanced Configuration*, this will show the contents of the Pound log. This is a very detailed log of all Pound SSL transactions.

SSL Termination (STunnel)

/var/log/stunnel.log

If activated via *Edit Configuration > SSL - Advanced Configuration*, this will show the contents of the STunnel log. The required debug level can also be set.

WAF

/var/log/httpd/modsec_audit.log
/var/log/httpd/modsec_debug.log
/var/log/httpd/modsec_audit_<WAF-NAME>.log

2 general ModSecurity logs are available - one for audit and the other for debug information. Additional WAF specific ModSecurity logs are also created for each WAF instance.

WAF Error

/var/log/httpd/error_<WAF-NAME>.log

A WAF error log is created for each WAF instance.

Heartbeat

/var/log/ha.log

The heartbeat log shows the status of the heartbeat daemons. Heartbeat is used whether configured as a single device or as a clustered pair. The log provides a detailed real-time status of heartbeat.

Apache Log

/var/log/httpd/user_access.log

Shows Apache user access logs. Can be generated by WebUI and the WAF (Web Application Firewall) since both utilize Apache for their operation.

Apache Error Log

File: /var/log/httpd/error.log

Shows Apache errors. These can be generated by the WebUI and by the WAF (Web Application Firewall).

Appliance 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 not listed.

Check Status

Virtual Service	Real Server	Forwarding Method	Weight	Active Connections	Inactive Connections
HTTP-Cluster1 192.168.110.120 port 80/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.

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 80/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.

Layer 4 traffic Counters

This report shows the volume of traffic to each Real Server since the counters were last re-set. If a Real Server has failed a health check, it will not be listed.

Virtual Service	Real Server	Connections	Incoming Packets	Outgoing Packets	Incoming Bytes	Outgoing Bytes
HTTP-Cluster1 192.168.110.120 port 80/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

These reports are generated in real time. Direct Routing is the default load balancing method and you will not see any stats for return packets as shown above (as they do not pass through the load balancer). They will be seen for NAT mode since return traffic does pass back via the load balancer.

In the example above, the details for RIP3 are not displayed because it's failing its health checks.

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.


```
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

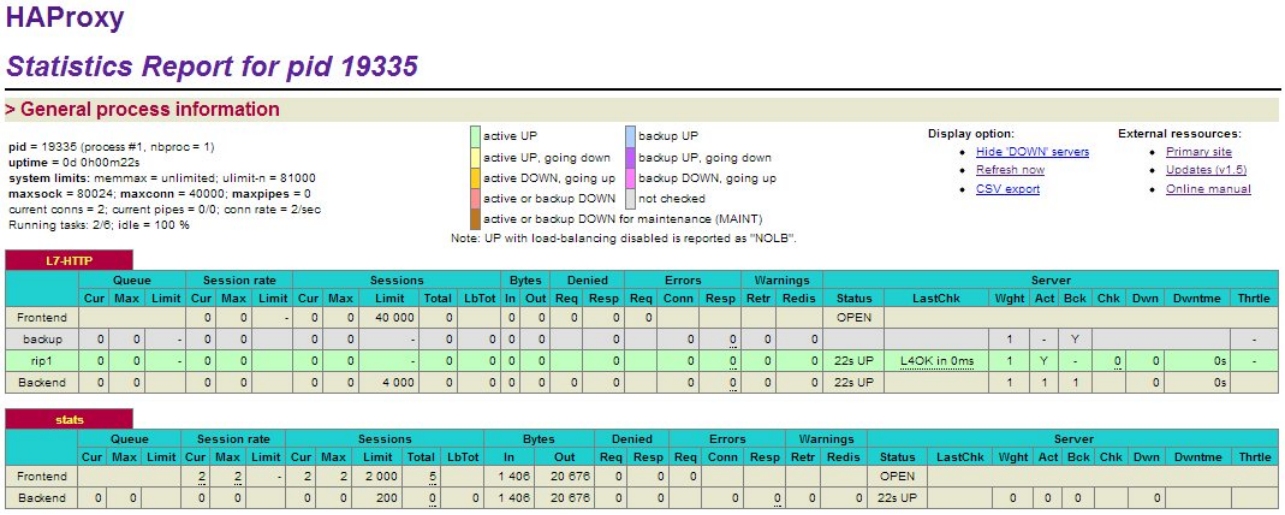
The IPVS connection entries in state **NONE** represent the persistence related entries for client connections, and are not actual client connections. These only appear when persistence is enabled.

Layer 4 Current Connections (Resolve Hostnames)

This is the same as the current connections report but is slower as it looks up the DNS name of each IP address.

Layer 7 Status

This report is provided by the stats instance of HAProxy. This web page contains the current live status of all configured layer 7 HAProxy Virtual Servers and Real Servers.



HAProxy

Statistics Report for pid 19335

> General process information

pid = 19335 (process #1, nbproc = 1)
uptime = 0d 0h00m22s
system limits: memmax = unlimited; ulimit-n = 81000
maxsock = 80024; **maxconn** = 40000; **maxpipes** = 0
current conns = 2; current pipes = 0/0; conn rate = 2/sec
Running tasks: 2/6; 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)
Note: UP with load-balancing disabled is reported as "NOLB".

Display option:
- Hide 'DOWN' servers
- Refresh now
- CSV export

External ressources:
- Primary site
- Updates (v1.5)
- Online manual

L7-HTTP

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	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					OPEN								
backup	0	0	-	0	0		0	0	-	0	0	0	0		0		0	0	0	0			1	-	Y				-
rip1	0	0	-	0	0		0	0	-	0	0	0	0		0		0	0	0	0	22s UP	L4OK in 0ms	1	Y	-	0	0	0s	-
Backend	0	0		0	0		0	0	4 000	0	0	0	0	0	0		0	0	0	0	22s UP		1	1	1		0	0s	

stats

	Queue			Session rate			Sessions					Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				2	2	-	2	2	2 000	5		1 406	20 676	0	0	0					OPEN								
Backend	0	0		0	0		0	0	200	0	0	1 406	20 676	0	0		0	0	0	0	22s UP		0	0	0		0		

Layer 7 Stick Table

Displays the layer 7 stick tables. For example, if a layer 7 VIP is created using RDP cookie persistence, a stick table will be used. The related VIP is then available in the drop-down as shown below:

Notes

1. Stick tables are used when either source IP persistence or RDP cookie persistence is used with layer 7 Virtual Services.
2. 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 Generic State

This report shows information about the running configuration of GSLB and also the health state of each member/endpoint.

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.

Graphing

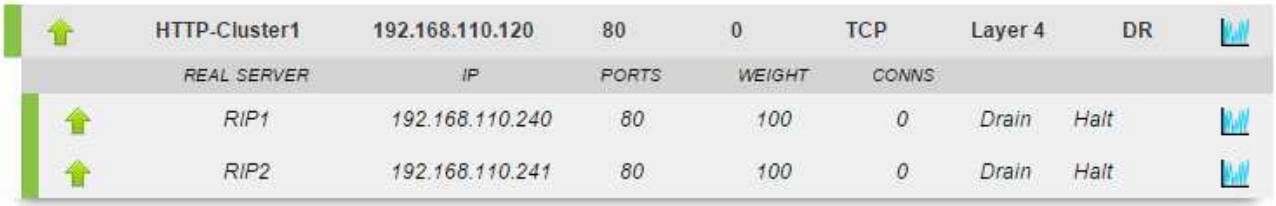
Graphs are automatically configured when new Virtual and Real Servers are defined.

Graphs - Load Balanced Services

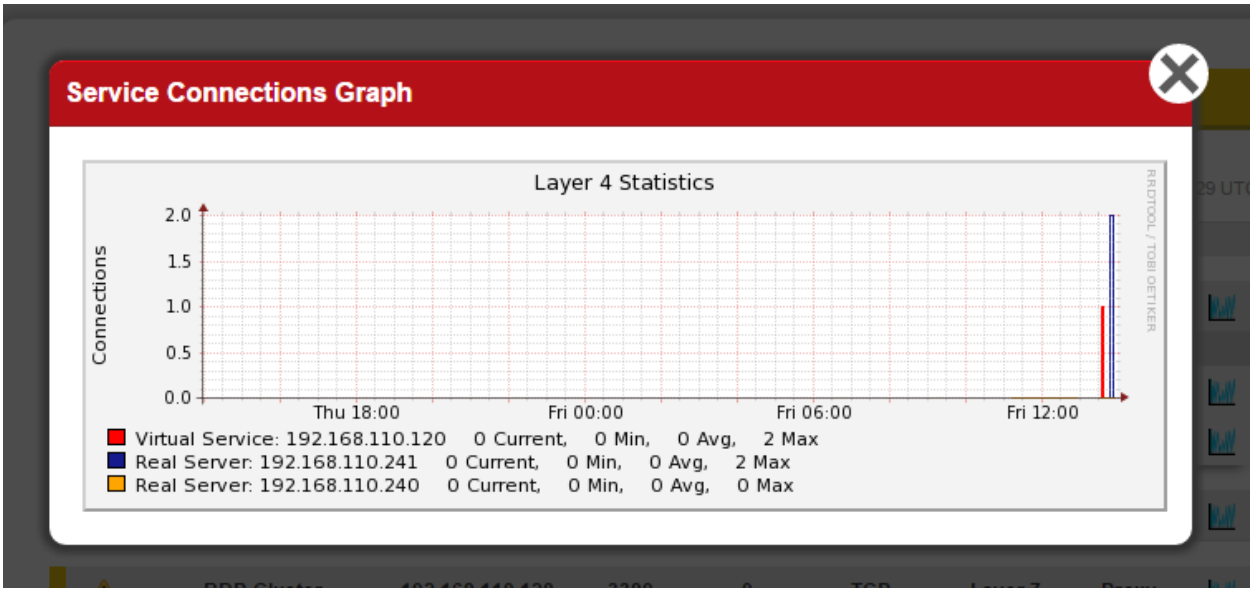
Graphs for the configured Virtual & Real Servers can be accessed either from the System Overview using the appropriate blue colored graph icon that appears next to each VIP and RIP or from the drop-down available in the WebUI under *Reports > Graphing*.

Using the System Overview:

The graph is displayed by clicking the relevant blue icon that's displayed next to each VIP/RIP:



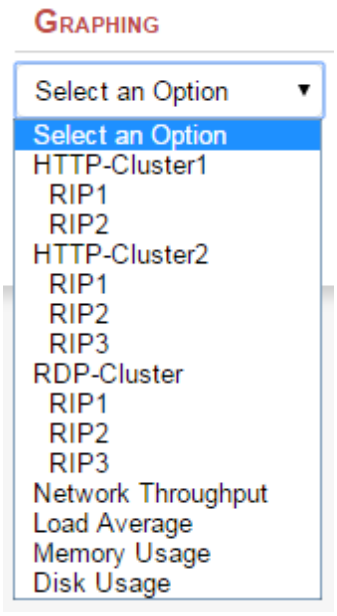
When this method is used, the daily Service Connections Graph (i.e. the last 24 hrs) is displayed for the particular VIP or RIP:



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:

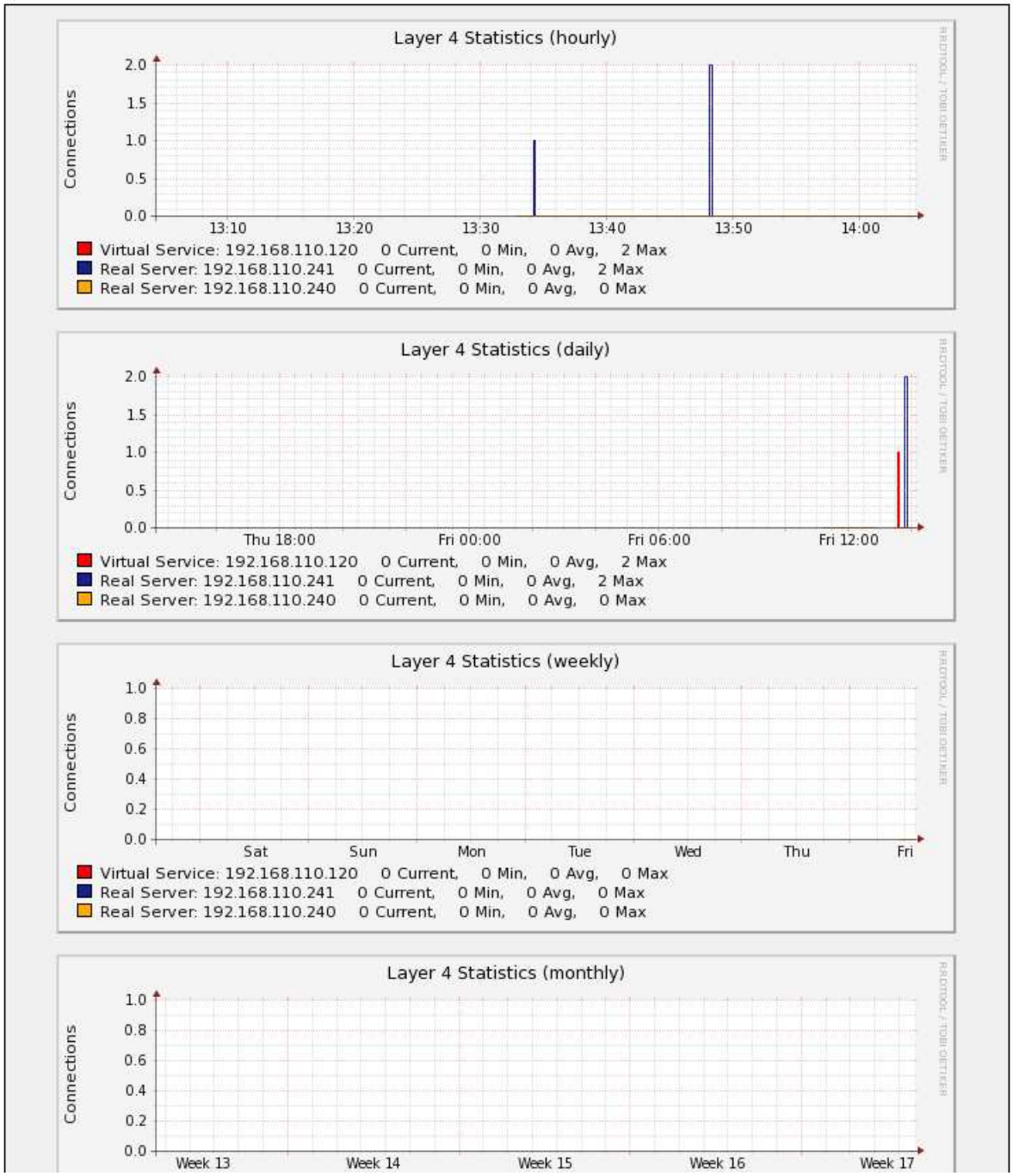
When selected, a drop-down similar to the following is displayed:



Note	As VIPs & RIPs are added or removed, these are automatically added/removed from the drop-down list.

Note	For more information on using the System Overview please refer to Chapter 8 - Real Server Health Monitoring & Control.

When selected in this way, a complete list of graphs is displayed for the VIP/RIP selected as shown below:



The following graphs are displayed for each VIP or RIP selected:

- 5 x **Connection graphs :** Hourly, daily, weekly, monthly and yearly
- 5 x **Bytes/s graphs** : Hourly, daily, weekly, monthly and yearly

Graphs - Appliance Specific

Appliance specific graphs are available for the following statistics:

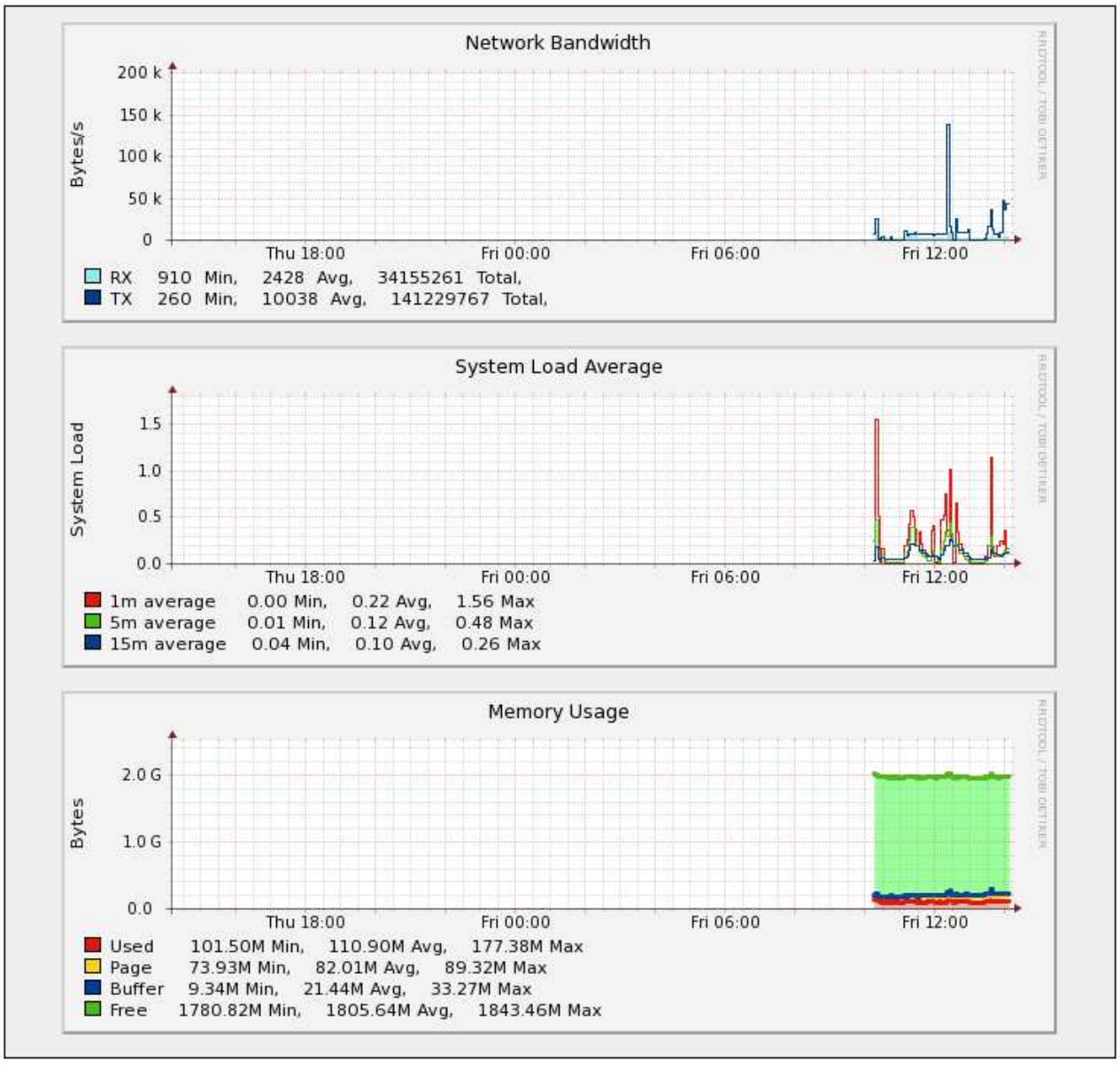
- Network Throughout
- Load Average

- Memory Usage
- Disk Usage

The first three graphs listed above are displayed in the System Overview by default although these can be disabled/hidden if preferred using the WebUI menu option: *Local Configuration > Graphing*.

All four graphs can also be accessed using the WebUI menu option: *Reports > Graphing*, then selecting the required graph from the bottom of the list.

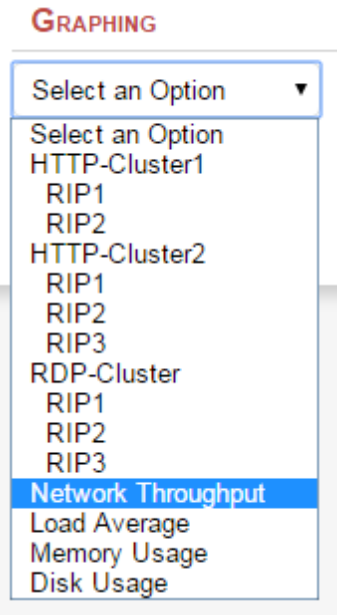
System Overview Graphs:



As shown above, daily graphs for **Network Bandwidth**, **System Load Average** and **Memory Usage** are displayed by default in the System Overview. Clicking anywhere within these graph opens the full list of related graphs (hourly, daily, weekly etc.). This is the same as selecting the graph in the Reports menu as explained below.

Using the Reports Menu

When selected, a drop-down including all VIPs/RIPs as well as the 4 appliance specific graphs is displayed:



Graph Options

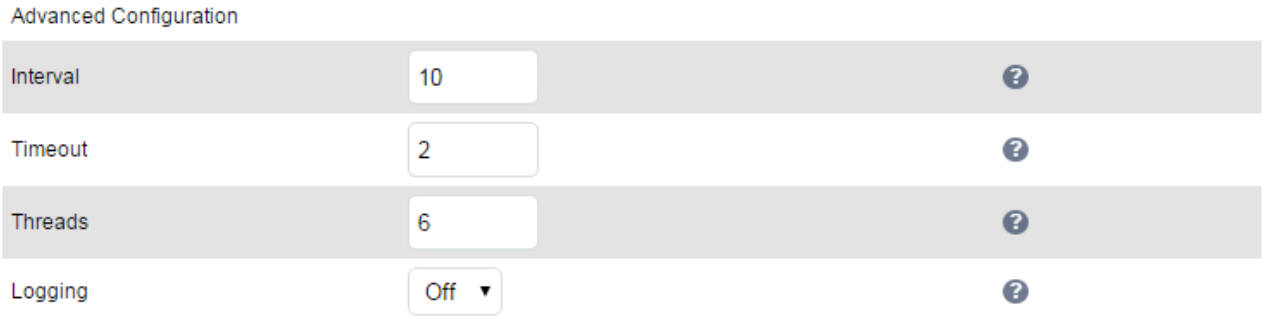
A number of graph options are available.

To change the settings:

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

- Data collection for each graphing category can be enabled (default) by selecting *On* and clicking **Update**
- Data collection for each graphing category can be disabled by selecting *Off* and clicking **Update**
- The stored data for each graphing category can be removed by selecting *Delete* and clicking **Update**

Advanced Configuration Settings



Interval - Set the data collector Interval time specified in seconds. Change the interval for which data is recorded by the collector. This is a global value and will effect all collectors. Do not change unless advised to do so by support. *WARNING - Changing this value will reset the RRD database files and you will loose all your previous data!!*

Timeout - Set the data collector timeout specified in seconds. Change the timeout for the data collector when querying the various services. Do not change unless advised to do so by support.

Threads - Set the number of data collector process threads. Change the number of collector process threads to use for reading stats. Do not change unless advised to do so by support.

Logging - Enable collector logging for collectd. Warning this is incredibly verbose and should only be used for debugging purposes.

SNMP Reporting

By default, SNMP is disabled on the appliance. For more 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 2 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..
2	wlc	tcp	192.168.111.222	80	undefined	300	255..

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.

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::l7FePxname.493413195 = STRING: VIP4
L7STAT-EXPERIMENTAL-MIB::l7FePxname.1593140907 = STRING: VIP3
```

The same can be achieved using the MIB object name 'l7FePxname':

```
[root@lbmaster ~]# snmpwalk -v 2c -c public localhost l7FePxname
```

Display the L7Frontend table:

```
[root@lbmaster ~]# snmptable -c public -v 2c localhost l7Frontend
```

l7FeIid	l7FePxname	l7FeStatus	l7FeMode	l7FeConnRate	l7FeConnRate...	l7FeConn
4	stats	nolb	http	0	2	3
493413195	VIP4	nolb	tcp	0	0	0
1593140907	VIP3	nolb	http	0	0	0

Display the L7Backtend table:

```
[root@lbmaster ~]# snmptable -c public -v 2c localhost l7Backend
```

l7BeIid	l7BePxname	l7BeStatus	l7BeWeight	l7BeMode	l7BeCookie	l7BeAlgo
4	stats	up	0	http		roundRobin
493413195	VIP4	up	100	tcp		leastConnectio
1593140907	VIP3	up	100	http	SERVERID	leastConnectio

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 'l7SrvSname':

```
[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...	l7SrvAddressI
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
```

Chapter 14 - 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    336 192.168.100.237:22      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
```

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.

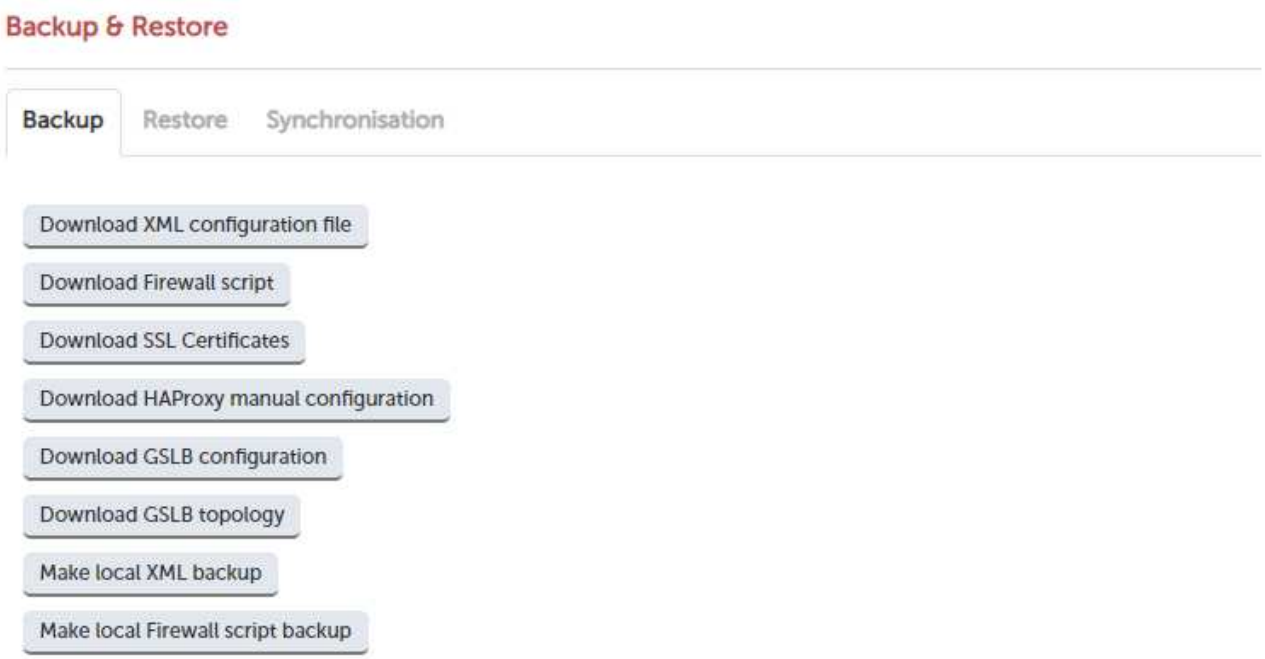
PuTTy

PuTTy is an open source SSH client for Windows. It can be downloaded from here.

Chapter 15 - Backup & Restore and Disaster Recovery

Backup & Restore

The WebUI can be used to perform backup and restore functions. To access these options use the WebUI menu option: *Maintenance > Backup & Restore*.



Backup Options

Download XML configuration file - download and save the load balancer's XML configuration file

Download Firewall script - download and save load balancer's firewall script

Download SSL Certificates - download and save the load balancer's SSL certificates as a compressed archive file

Note	This option should be used in conjunction with the XML restore option to ensure the certificates are uploaded and correctly recreated in the WebUI.

Download HAProxy manual configuration - download and save the load balancer's layer 7 manual configuration file

Download GSLB configuration - download the load balancer's GSLB configuration file

Download GSLB topology - download the load balancer's GSLB topology file

Make local XML backup - creates a local backup of the current XML file in /etc/loadbalancer.org/userbkup

Make local Firewall Script backup - creates a local backup of the current rc.firewall in /etc/loadbalancer.org/userbkup

Restore Options

Upload XML file and Restore - upload an XML file and restore load balancer settings

Upload and Restore SSL Certificates - upload and restores an SSL certificate compressed archive file that was created using the *Download SSL Certificates* backup option

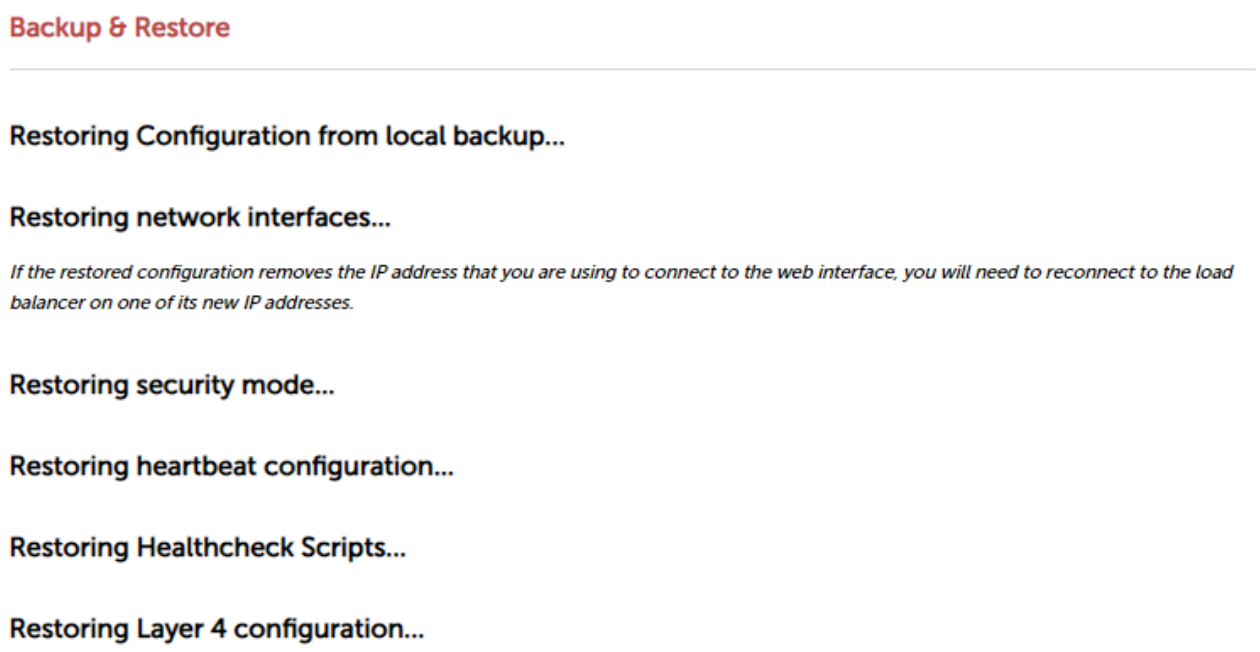
Restore from the last local XML backup - Restore the last local backup created with the 'Make local XML Backup' option

Restore Manufacturer's defaults - Restore system settings to default values

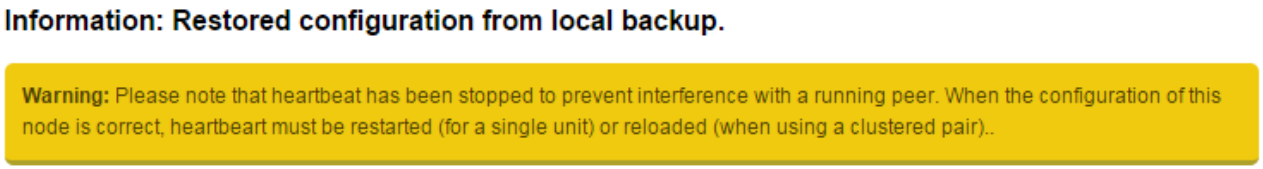
Note	The XML restore feature is not backward compatible with previous major versions of the software, e.g. it's not possible to restore a V.6.4 XML file to a v8.5.x appliance.

XML File Restore Process

The screen shot below shows an ongoing restore from a local XML file backup:



Once complete, you'll need to either restart or reload heartbeat to complete the restore process as explained in the yellow message box:



Disaster Recovery

To recover a single appliance, you'll need to restore your configuration files. 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. For more information, 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 a backup of the XML file as well as other relevant configuration files and keep them stored in a secure location off the load balancer. Ideally you should keep a backup of both the Primary and Secondary configurations. This can easily be done by following the steps below:

Backing Up Configuration Files to a Remote Location

Login to the WebUI:

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

Backup the XML configuration file:

1. Select *Maintenance > Backup & Restore* and click **Download XML configuration file**.
2. Save the file to an appropriate location.

If you have any manually defined firewall marks or you have made any other changes to the firewall script, backup the firewall configuration:

1. Select *Maintenance > Backup & Restore* and click **Download Firewall Script**.
2. Save the file to an appropriate location.

If you're terminating SSL on the load balancer, backup your certificates:

1. Select *Maintenance > Backup & Restore* and click **Download SSL Certificates**.
2. Save the file to an appropriate location.

If you have any manually defined layer 7 services, back these up:

1. Select *Maintenance > Backup & Restore* and click **Download HAProxy manual configuration**.
2. Save the file to an appropriate location.

If you have any manually defined WAF services, back these up:

1. Select *Cluster Configuration > WAF - Manual Configuration*.
2. Copy your custom WAF definition to an appropriate location.

If you have configured GSLB, back this up:

1. Select *Maintenance > Backup & Restore* and click **Download GSLB configuration**.
2. Save the file to an appropriate location.

If you have configured GSLB topology, back this up:

1. Select *Maintenance > Backup & Restore* and click **Download GSLB Topology**.

2. Save the file to an appropriate location.

Using wget to Copy the Files

It's also possible to use wget from a remote Linux host to pull the XML configuration file and firewall script from the appliance:

```
wget --no-check-certificate --user=loadbalancer --password=loadbalancer https://<appliance-IP>:9443/lbadmin/config/getxmlconfig.php -O lb_config.xml
```

```
wget --no-check-certificate --user=loadbalancer --password=loadbalancer https://<appliance-IP>:9443/lbadmin/config/getfirewall.php -O rc.firewall
```

Note	Replace 'loadbalancer' with the password for your appliance.

Restoring Files to the Appliance

Login to the WebUI:

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

Restore the XML configuration file:

1. Select *Maintenance > Backup & Restore* and select the *Restore* tab.
2. Click the **Choose file** button and browse to and select the relevant XML file.
3. Click **Upload**.

If your configuration includes SSL terminations, restore your certificates:

1. Select *Maintenance > Backup & Restore* and select the *Restore* tab.
2. Click the **Choose file** button and browse to and select the relevant tar.gz file.
3. Click **Upload**.

Firmware Recovery using a USB Memory Stick

If a hardware appliance has suffered a hardware failure that requires the firmware to be re-installed, follow steps 1 to 5 below.

Note	This will only work on 64Bit hardware. From v6.x onward, all appliances are 64Bit. If you're running an older version, this may or may not be possible depending on the hardware.

Step 1 - Check older hardware for Compatibility

If you are still running v5.x and wish to determine whether your appliance is 64Bit and can be upgraded to the latest version, use the following command:

```
grep flags /proc/cpuinfo
```

This can be run using the WebUI menu option: *Local Configuration > Execute Shell command*, at the console or via a terminal session.

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**.

If **lm** (long mode) is present in the output then the CPU is 64Bit and you can proceed. If not then your appliance is 32Bit and you are limited to the latest v5 software.

Note	For further assistance to determine if your appliance can be upgraded to the latest version please contact support@loadbalancer.org.

Step 2 - Download the latest appliance disk image

The latest disk image can be downloaded from our website - please contact support for more information.

Step 3 - 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 4 - Prepare the USB stick

Under Linux

After formatting the USB stick run the following command:

```
dd if=/imagefilename.img of=/dev/nameofusbdisk
```

e.g.

```
dd if=/tmp/v7.5.0_r3368.img of=/dev/sda
```

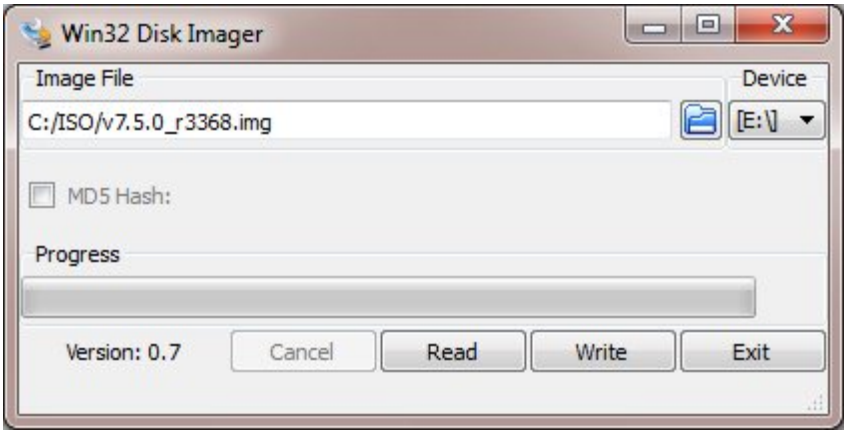
Do not use /dev/sdax where 'x' is a number, for example - /dev/sda1 as this will install to a partition on your usb stick. Use the whole disk **/dev/sda** Instead.

Be careful using this command - make sure you specify the correct disk !!

Under Windows

For Windows, a third party image writer must be used. Several free options are available, the example below uses ***Win32 Disk Imager*** which can be downloaded here.

First extract the archive, then run the executable to install the application, then run the application.



Select the image file and set the appropriate output Device as shown above.

Click **Write**.

Be careful using this command - make sure you specify the correct disk !!

Step 5 - Restore the appliance using the USB stick

1. Change the appliance's BIOS settings to boot from USB first (on some models the stick must be plugged in to allow it to be selected as a boot device).
2. 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>
The installation will take around 2-3 minutes, once complete the following message will be
displayed:
Installation Finished
```

3. As directed, press any key to shutdown the load balancer.
4. Once shutdown, remove the USB stick.
5. Power up the appliance.
6. Login at the console:

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

Note	Console and SSH password access are disabled by default. These options can be enabled using the WebUI option: *Local Configuration > Security.* Set *Appliance Security Mode* to **Custom** and enable the required option(s).

7. Run the following command:

```
lbrestore <ENTER>
```

8. Reboot the appliance once again.

9. Set the required IP address using the Network Setup Wizard as described here.

Note

You'll need to reapply your license key file to ensure the newly restored appliance is correctly licensed. Please contact support@loadbalancer.org 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.

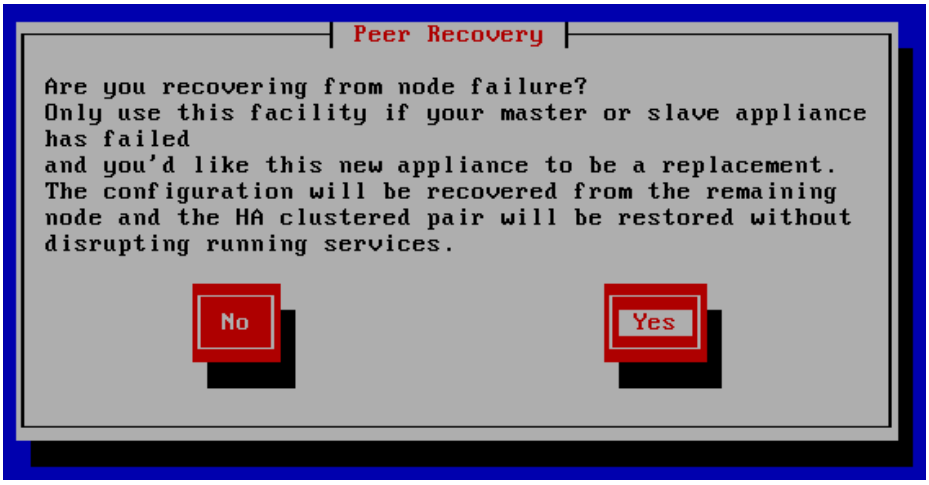
Note

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.

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:

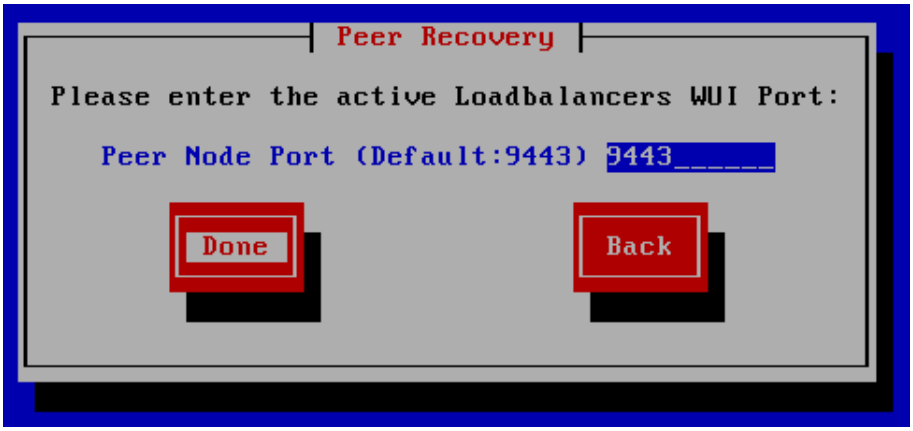
   username: setup
   password: setup

5. Now run through the Network Setup Wizard to configure the initial network settings - ensure these are the same as the failed appliance.
6. Once the initial network settings have been configured, you'll be asked if you're recovering from node failure as shown below:



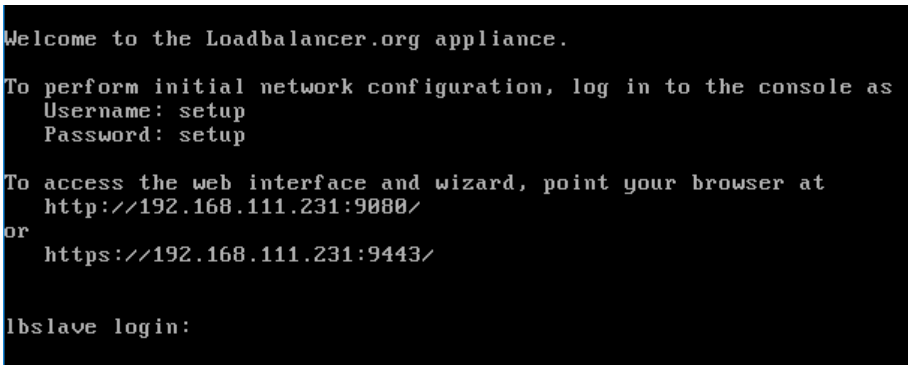
7. At this point, select **Yes** and hit <ENTER>.
8. You'll now be prompted for information about the remaining (still working) appliance as shown below:

9. Enter the IP address of the remaining appliance, select **Done** and hit <ENTER>.



10. Enter the WebUI port of the remaining appliance, select **Done** and hit <ENTER>.

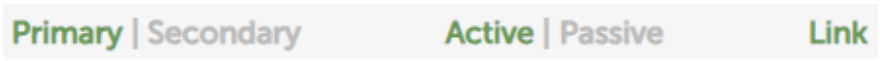
11. Enter the WebUI password for the 'loadbalancer' user on the remaining appliance, select **Done** and hit <ENTER>.



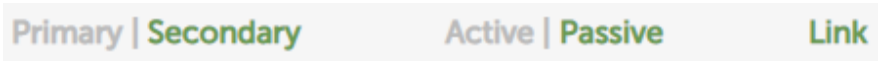
12. Once the restore completes, the login prompt will be displayed.
13. To complete the process, reboot the newly restored appliance.

Once rebooted, the HA pair will be re-synchronized and fully recovered

The Primary's WebUI should now display:



The Secondary's WebUI should now display:



Chapter 16 - 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/)

WebUI Support Options

The WebUI's Support menu option includes 3 sub-options that are provided to assist when support is required.

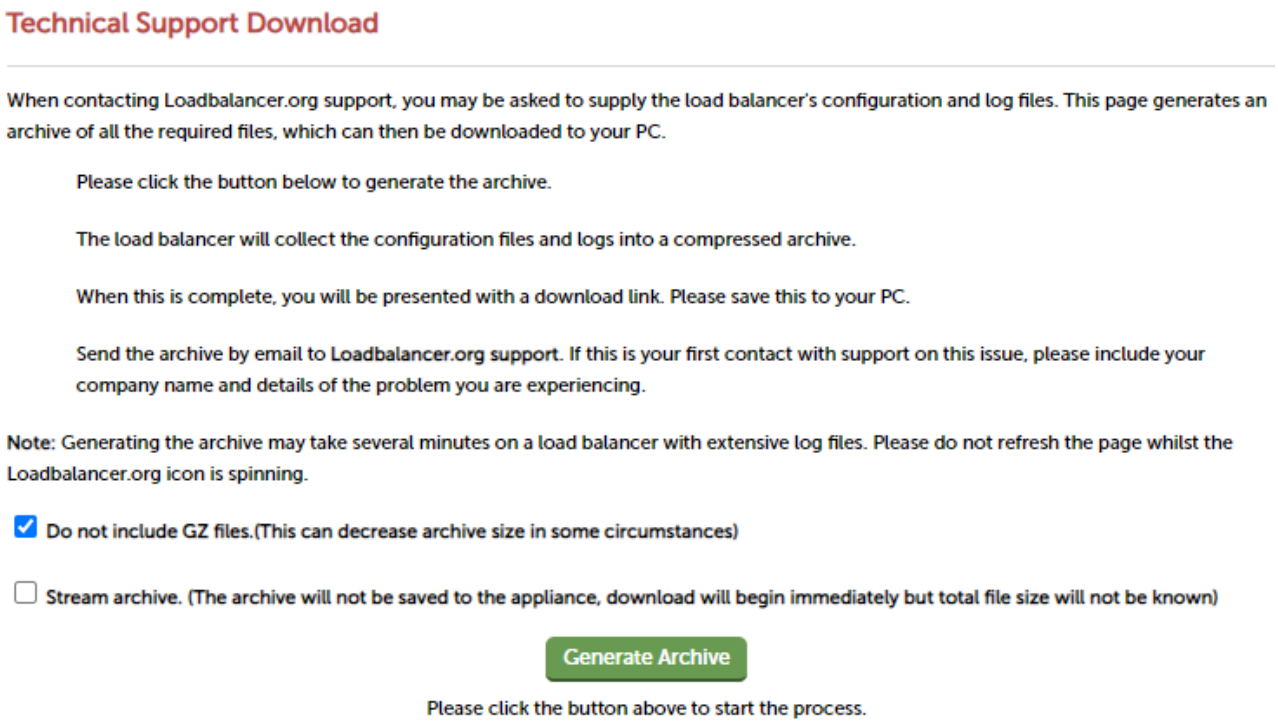
Contact Us

This option provides details on how to contact Loadbalancer.org, how to report any issues and what information we'll need to resolve issues as quickly as we can. As mentioned, the Loadbalancer.org support team can be contacted using the email address: support@loadbalancer.org.

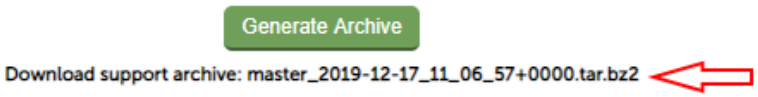
Sending an email to this address creates a ticket in our help desk system and enables all technical support staff to view the case. This is the most efficient way to contact support and guarantees that any reported issues will be acted upon and addressed as quickly and efficiently as possible.

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.



- 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 check-box shown above
- To stream the archive drectly to your browser's download location rather than saving it to the appliance, enable (check) the *Stream Archive* check-box
- To generate the archive, click the **Generate Archive** button
- Once complete, a link will be available to download the archive:



Once downloaded, attach the file to your email when contacting support, or if the file is large, it can be posted to our upload server - please ask our support staff about this option.

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 checked (enabled).
3. Click Update.

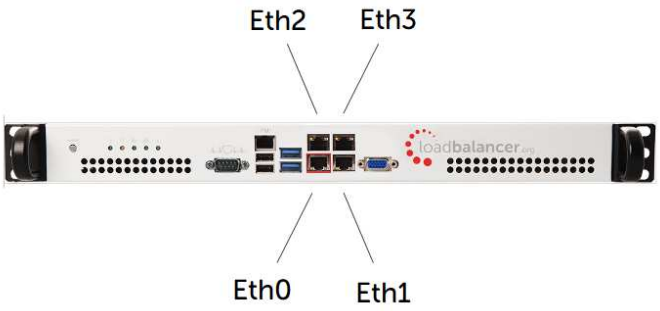
To invoke Online Chat:

1. Using the WebUI, navigate to: *Live Chat*.
2. Enter your details and the question you have.

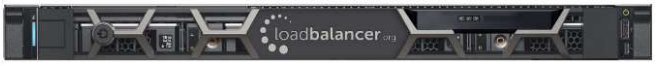
Appendix

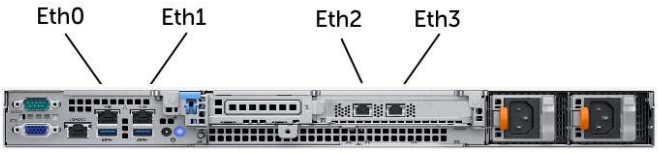
Front & Rear Panel Layouts

Enterprise 1G

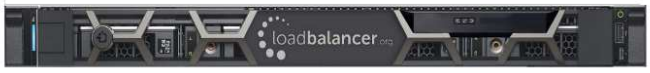


Enterprise 10G/25G/40G/50G

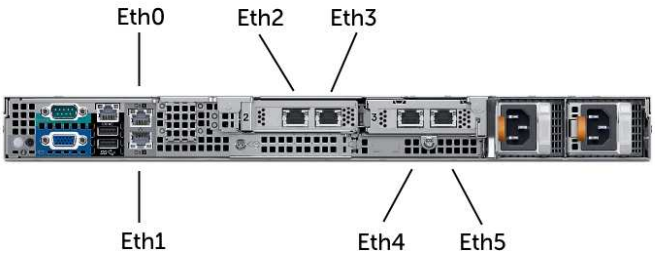




Enterprise 100G



* The number of interfaces depends on your choice of interface cards



IPMI (Remote Management) Configuration

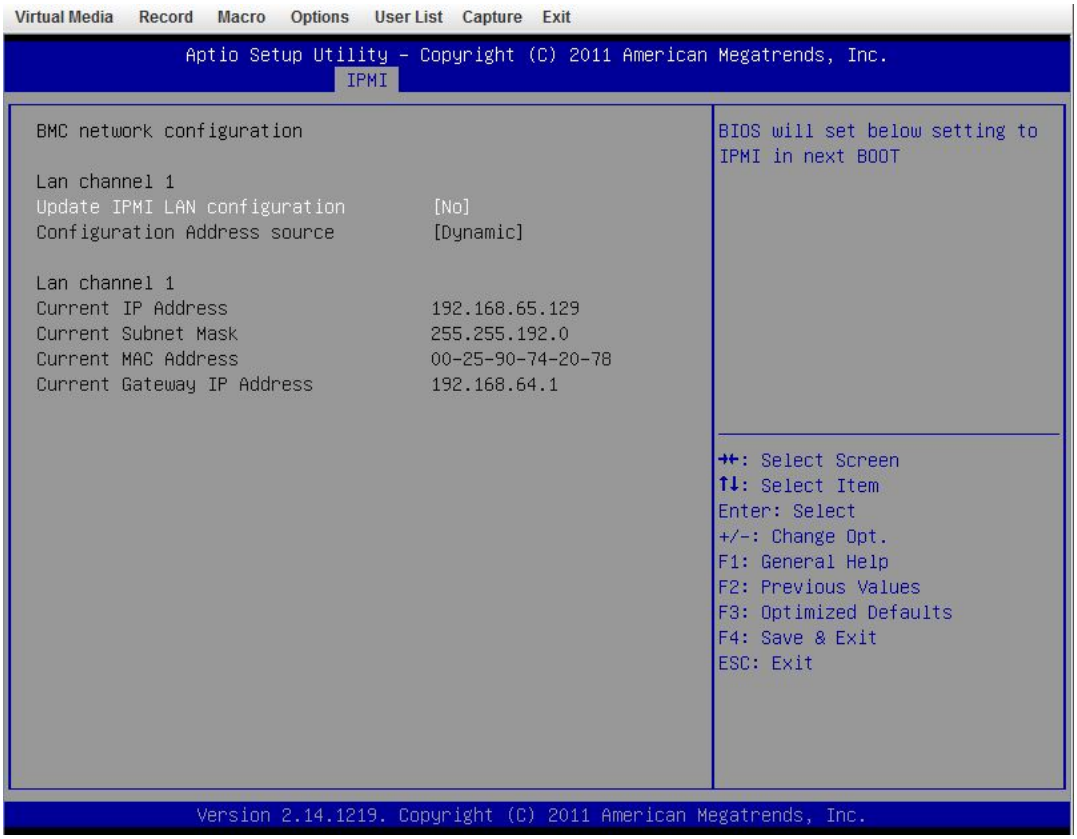
The Enterprise 1G supports IPMI to allow remote control & management. This can either be accessed via the dedicated IPMI Ethernet interface or via one of the standard Ethernet interfaces in bridged mode.

To use the dedicated IPMI interface, ensure that a network cable is plugged into the interface before powering up the appliance.

Configuring the IP Address

By default the IP address is set using DHCP. The address allocated is displayed in the IPMI sub-menu in system setup. If preferred, a static IP address can also be set using the same menu. To access system setup, hit <DEL> as directed at boot time.

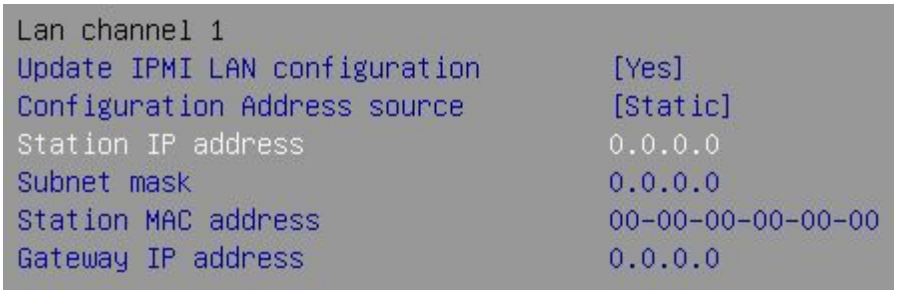
IPMI BIOS Menu:



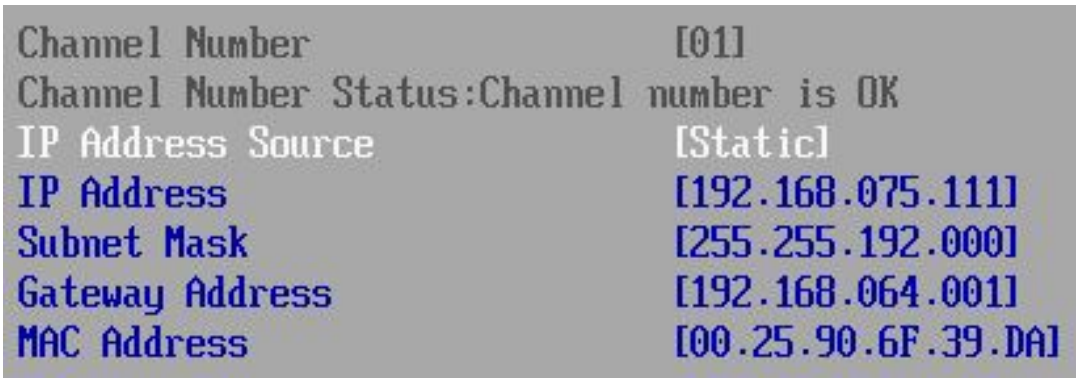
To set the address:

Change **Update IPMI LAN configuration** to 'Yes'
Change **Configuration Address Source** to 'Static'

Now set the IP address, mask etc. as required.



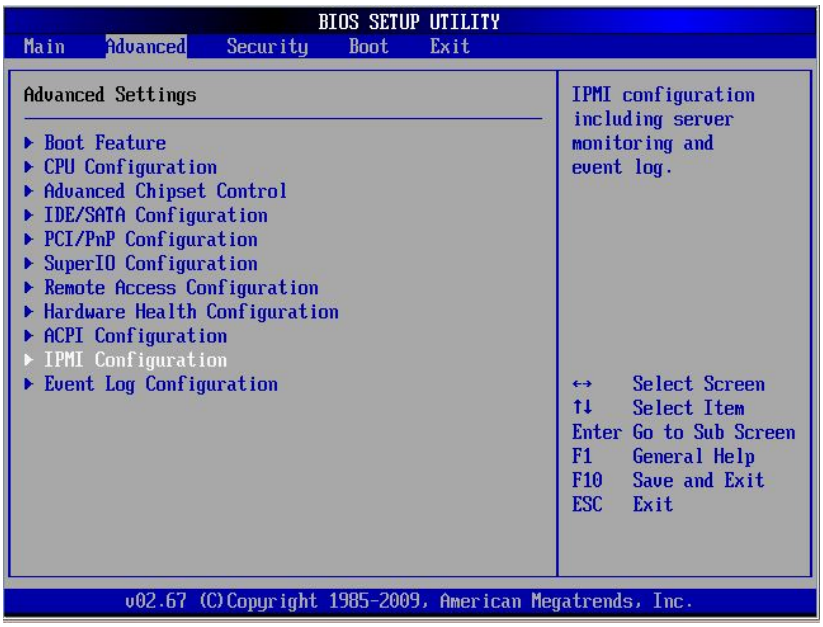
IPMI BIOS Menu:



To set the address:

Select **Set LAN Configuration**.

Change **IP Address Source** to 'Static'
Now set the IP address, mask etc. as required.



Accessing the login page:

Using a browser, connect to **http://<ip address>**

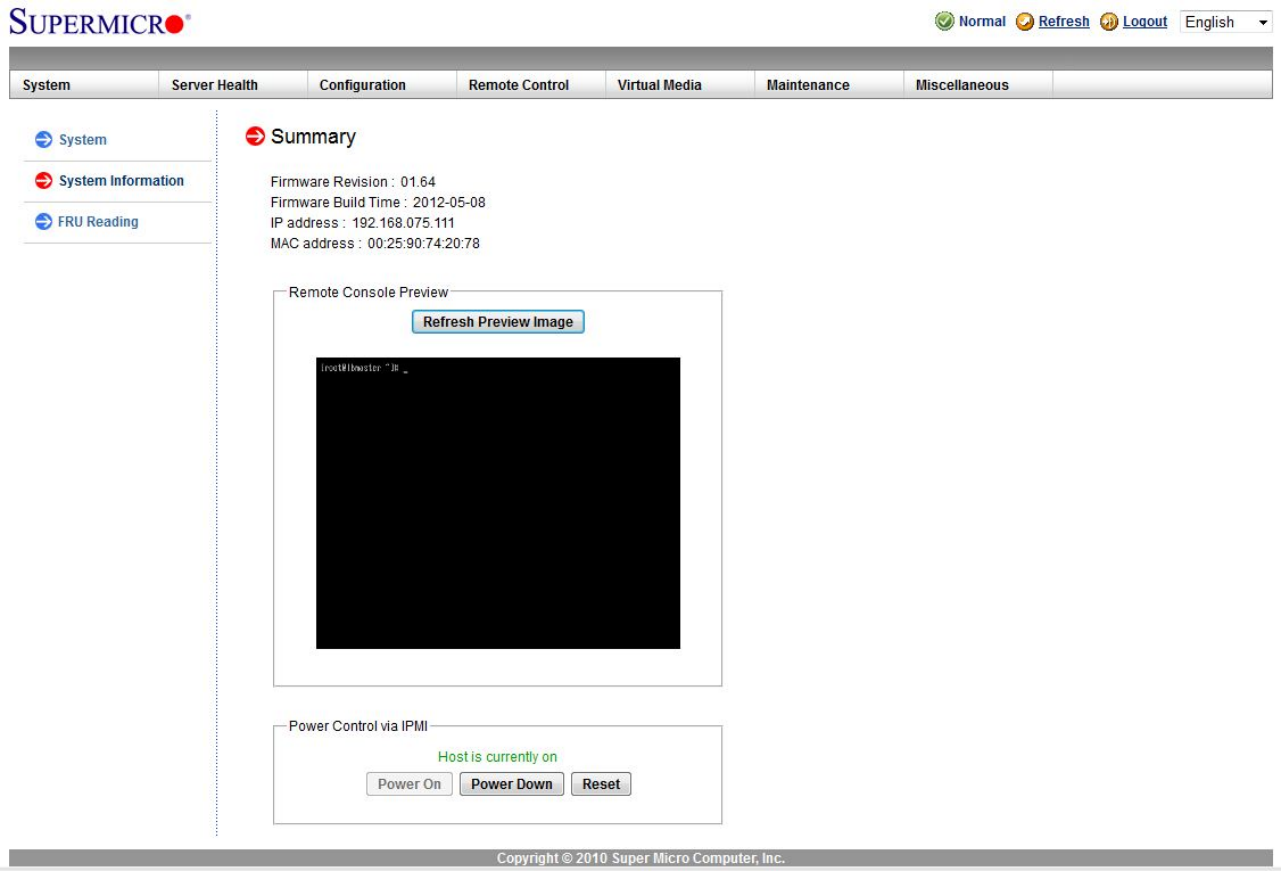
the following login prompt is displayed:

username: **ADMIN**

default password: **ADMIN**

Note	For certain appliances the IPMI password may have been updated. If this is the case the updated password will be noted on a label applied on the top cover of the appliance. If you have any issues please contact support@loadbalancer.org.

Once logged in, the following screen is displayed:



IPMI Interface

As mentioned above IPMI can be accessed via the dedicated interface or via one of the standard on-board NICs. This can be configured in the IPMI interface using: *Configuration > Network > LAN Interface*.

- **Dedicate** - use the dedicated interface only
- **Share** - run in bridge mode using one of the standard NICs
- **Failover** - allows either connection method to be used (the default)

Remote Control

To access the systems console, simple click on the Remote Console Preview image. A new window will open with access to the console of the appliance.

Note

You cannot SSH into the module directly. You need to connect via the IPMI's web interface, then use the remote control option as mentioned above. This can also be accessed using the 'Remote Control' option in the top menu. From here you can use the Launch Console option to launch a virtual Java console which will allow you to use the device as if you stood in front of the device. Next the 'Power Control' options menu will give you several options such as Restart Server, Power off and Power Cycle server. these options will perform the same function as pressing the physical reset button on the unit (Reset Server) as well as being able to perform the same functions as the physical power switch as well.

Please do remember that the IPMI power control options are completely independent of the Loadbalancer software

and that the reset option is the same as pressing reset on your PC.

iDRAC (Remote Management) Configuration

iDRAC is supported on the Enterprise 10G, Enterprise 50G and Enterprise 100G appliances.

> **Note** | For more information on iDRAC licensing options, please contact sales.

iDRAC IP Address

The Default IP address is **192.168.0.120/255.255.255.0**. This can be changed in a number of ways. The recommended method is to use the **iDRAC Setting Utility**. All methods are described here.

Logging in to iDRAC

The default user credentials are:

```
Username: root
Password: calvin
```

Details on how to change the default password are available here.

iDRAC Password Reset

If you've forgotton your iDRAC password, it can be reset as described here.

iDRAC Licensing

If required, trial iDRAC licenses are available here.

Details on importing your iDRAC license are available here.

More Information

The full iDRAC9 user guide is available here.

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
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

Mask	IP/CIDR Prefix
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

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.

United Kingdom

Loadbalancer.org Ltd.
Compass House, North Harbour
Business Park, Portsmouth, PO6 4PS
UK:+44 (0) 330 380 1064
sales@loadbalancer.org
support@loadbalancer.org

Canada

Loadbalancer.org Appliances Ltd.
300-422 Richards Street, Vancouver,
BC, V6B 2Z4, Canada
TEL:+1 866 998 0508
sales@loadbalancer.org
support@loadbalancer.org

United States

Loadbalancer.org, Inc.
4550 Linden Hill Road, Suite 201
Wilmington, DE 19808, USA
TEL: +1 833.274.2566
sales@loadbalancer.org
support@loadbalancer.org

Germany

Loadbalancer.org GmbH
Tengstraße 2780798,
München, Germany
TEL: +49 (0)89 2000 2179
sales@loadbalancer.org
support@loadbalancer.org