

Load Balancing RabbitMQ

Version 1.3.0



Table of Contents

1. About this Guide	4
2. Loadbalancer.org Appliances Supported	4
3. Software Versions Supported	4
3.1. Loadbalancer.org Appliance	4
3.2. RabbitMQ	4
4. RabbitMQ	4
5. Load Balancing RabbitMQ	4
5.1. Persistence (aka Server Affinity)	4
5.2. Virtual Service (VIP) Requirements	5
5.3. Server Feedback Agent	5
6. Deployment Concept	5
7. Load Balancer Deployment Methods	5
7.1. Layer 4 DR Mode	6
7.2. Layer 7 SNAT Mode	6
7.3. Our Recommendation	7
8. Configuring RabbitMQ for Load Balancing	8
8.1. Server Configuration	8
8.2. Layer 4 DR Mode – Solving the ARP Problem	8
9. Loadbalancer.org Appliance – the Basics	8
9.1. Virtual Appliance	8
9.2. Initial Network Configuration	8
9.3. Accessing the Appliance WebUI	9
Main Menu Options	10
9.4. Appliance Software Update	11
Determining the Current Software Version	11
Checking for Updates using Online Update	11
Using Offline Update	11
9.5. Ports Used by the Appliance	12
9.6. HA Clustered Pair Configuration	13
10. Appliance Configuration for RabbitMQ – Using Layer 4 DR Mode	13
10.1. Configuring the Virtual Service (VIP)	13
10.2. Defining the Real Servers (RIPs)	13
11. Appliance Configuration for RabbitMQ – Using Layer 7 SNAT Mode	14
11.1. Configuring the Virtual Service (VIP)	14
11.2. Defining the Real Servers (RIPs)	14
11.3. Finalizing the Configuration	15
12. Testing & Verification	15
12.1. Using System Overview	15
13. Technical Support	16
14. Further Documentation	16
15. Appendix	17
15.1. Server Feedback Agent	17
Windows Agent	17
Linux/Unix Agent	19
Custom HTTP Agent	20
15.2. Configuring HA - Adding a Secondary Appliance	20
Non-Replicated Settings	21
Adding a Secondary Appliance - Create an HA Clustered Pair	21

15.3. Solving the ARP Problem	23
Solving the ARP Problem for Linux	23
Windows Server 2012 & Later	29
16. Document Revision History	35

1. About this Guide

This guide details the steps required to configure a load balanced RabbitMQ environment utilizing Loadbalancer.org appliances. It covers the configuration of the load balancers and also any RabbitMQ configuration changes that are required to enable load balancing.

For more information about initial appliance deployment, network configuration and using the Web User Interface (WebUI), please also refer to the [Administration Manual](#).

2. Loadbalancer.org Appliances Supported

All our products can be used for load balancing RabbitMQ. For full specifications of available models please refer to <https://www.loadbalancer.org/products>.

Some features may not be supported in all cloud platforms due to platform specific limitations, please check with Loadbalancer.org support for further details.

3. Software Versions Supported

3.1. Loadbalancer.org Appliance

- V8.3.8 and later

Note

The screenshots used throughout this document aim to track the latest Loadbalancer.org software version. If using an older software version, note that the screenshots presented here may not match the WebUI exactly.

3.2. RabbitMQ

- All versions

4. RabbitMQ

RabbitMQ is an open source message broker. It uses a publish-subscribe model to route data from publishers to consumers. It is scalable and can be load balanced, acting as a reliable and highly available intermediary. It has support for management and monitoring, and has a range of tools and plugins available.

5. Load Balancing RabbitMQ

Note

It's highly recommended that you have a working RabbitMQ environment first before implementing the load balancer.

5.1. Persistence (aka Server Affinity)

RabbitMQ does not require session affinity at the load balancing layer by default.



5.2. Virtual Service (VIP) Requirements

To provide load balancing and HA for RabbitMQ, one virtual service is required.

The virtual service must be set to listen on the same port as the RabbitMQ service, which listens on port 5672 by default.

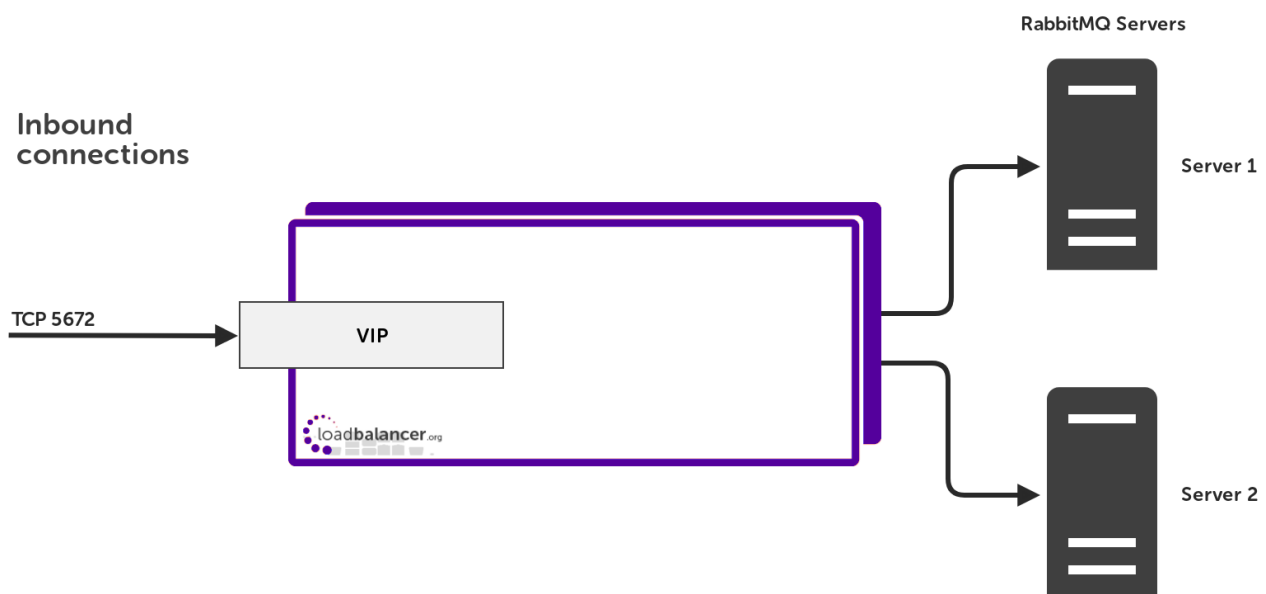
Our recommended configuration uses a layer 4 DR mode VIP. Using a layer 7 SNAT mode VIP is also supported.

5.3. Server Feedback Agent

It may be useful to adjust how much traffic is passed to the RabbitMQ servers depending on their CPU load. This can be done by installing the Loadbalancer.org server feedback agent on each RabbitMQ server and then re-configuring the Virtual Service to make use of the agent. The feedback agent is available for both Linux and Windows servers.

Please refer to [Server Feedback Agent](#) for full details on installing and configuring the server feedback agent.

6. Deployment Concept



VIPs = **V**irtual **I**P Addresses

Note

The load balancer can be deployed as a single unit, although Loadbalancer.org recommends a clustered pair for resilience & high availability. Please refer to [Configuring HA - Adding a Secondary Appliance](#) for more details on configuring a clustered pair.

7. Load Balancer Deployment Methods

The load balancer can be deployed in one of 4 fundamental ways: **Layer 4 DR mode**, **Layer 4 NAT mode**, **Layer 4 SNAT mode**, or **Layer 7 SNAT mode**. For RabbitMQ, layer 4 DR mode is recommended. Layer 7 SNAT mode is also supported. Both supported modes are described below.

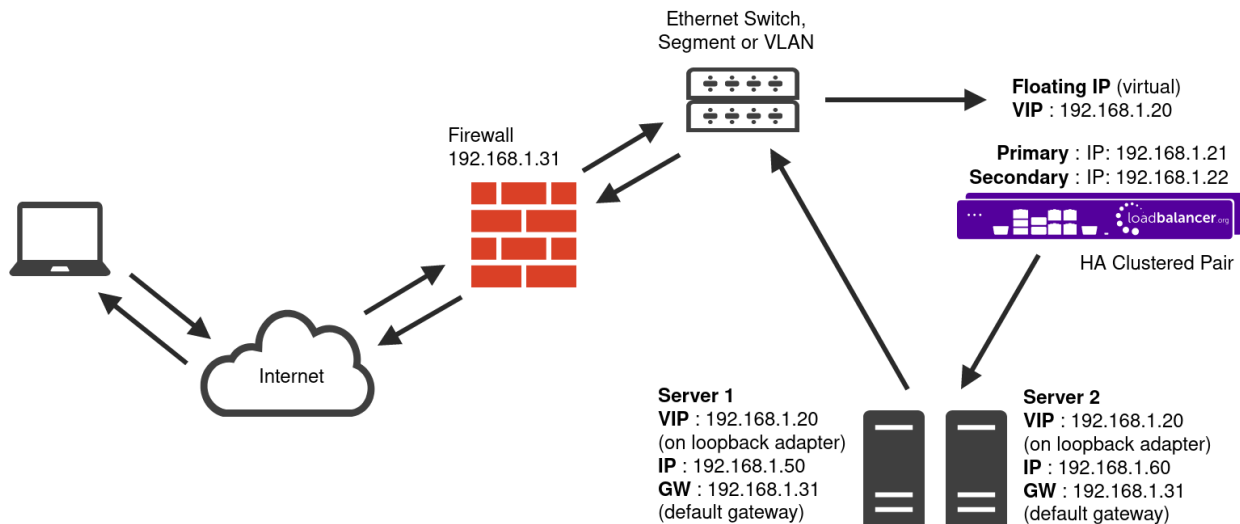


7.1. Layer 4 DR Mode

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 *nPath*.

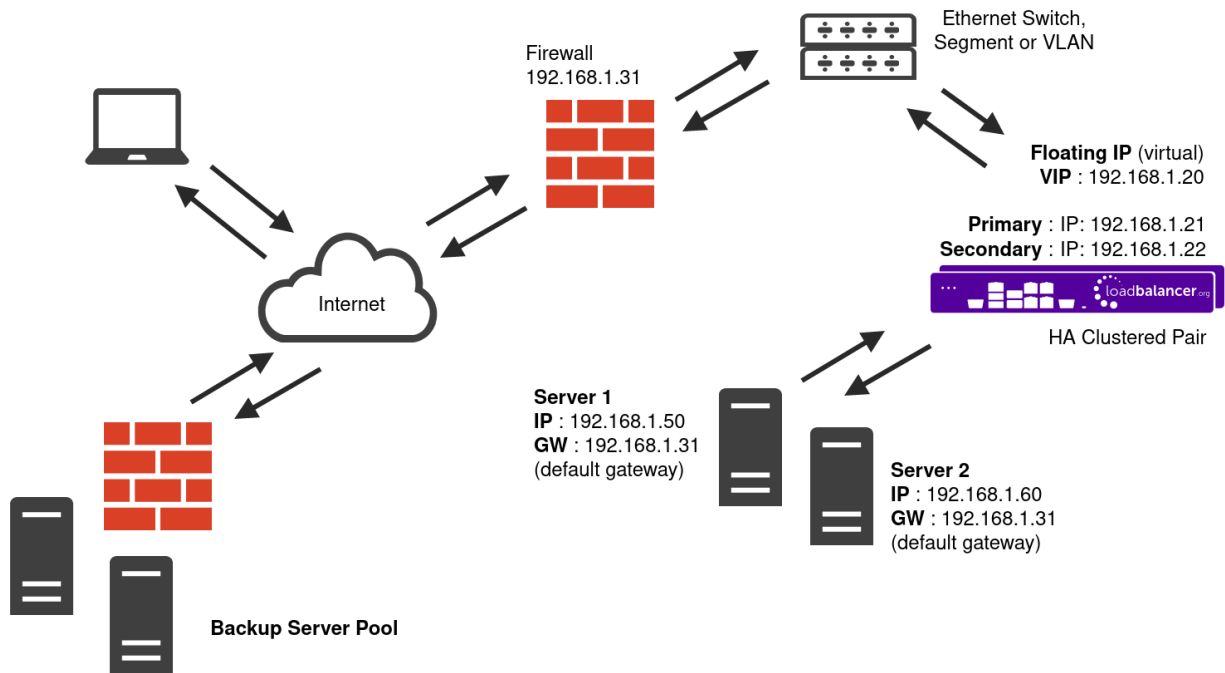


- DR mode works by changing the destination MAC address of the incoming packet to match the selected Real Server on the fly which is very fast.
- When the packet reaches the Real Server it expects the Real Server to own the Virtual Services IP address (VIP). This means that you need to ensure that the 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.

7.2. 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 mode-specific configuration changes to the load balanced Real Servers.
- Port translation is possible with Layer 7 SNAT mode, e.g. VIP:80 → RIP:8080 is supported.
- You should not use the same RIP:PORT combination for layer 7 SNAT mode VIPs and layer 4 SNAT mode VIPs because the required firewall rules conflict.

7.3. 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.

If DR mode cannot be used, for example if the real servers are located in remote routed networks, then SNAT mode is recommended.

If the load balancer is deployed in AWS or Azure, layer 7 SNAT mode must be used as layer 4 direct routing is not currently possible on these platforms.

8. Configuring RabbitMQ for Load Balancing

8.1. Server Configuration

RabbitMQ servers need to be configured for load balancing and high availability. This configuration is specific to the RabbitMQ service and is beyond the scope of this deployment guide.

Please refer to the following documentation on the RabbitMQ website which details the configuration that is required: <https://www.rabbitmq.com/ha.html>

8.2. Layer 4 DR Mode – Solving the ARP Problem

If using layer 4 DR mode, the 'ARP problem' must be solved on each real server for DR mode to work. For detailed steps on solving the ARP problem for Linux and Windows, please refer to [Solving the ARP Problem](#) for more information.

For a detailed explanation of DR mode and the nature of the ARP problem, please refer to [Layer 4 DR Mode](#).

9. Loadbalancer.org Appliance – the Basics

9.1. Virtual Appliance

A fully featured, fully supported 30 day trial is available if you are conducting a PoC (Proof of Concept) deployment. The VA is currently available for VMware, Virtual Box, Hyper-V, KVM, XEN and Nutanix AHV and has been optimized for each Hypervisor. By default, the VA is allocated 2 vCPUs, 4GB of RAM and has a 20GB virtual disk. The Virtual Appliance can be downloaded [here](#).

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

Please refer to [Virtual Appliance Installation](#) and the ReadMe.txt text file included in the VA download for additional information on deploying the VA using the various Hypervisors.


Note

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

9.2. Initial Network Configuration


After boot up, follow the instructions on the appliance console to configure the management IP address, subnet mask, default gateway, DNS Server and other network settings.




 **Important** Be sure to set a secure password for the load balancer, when prompted during the setup routine.

9.3. Accessing the Appliance WebUI


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

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

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

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


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

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

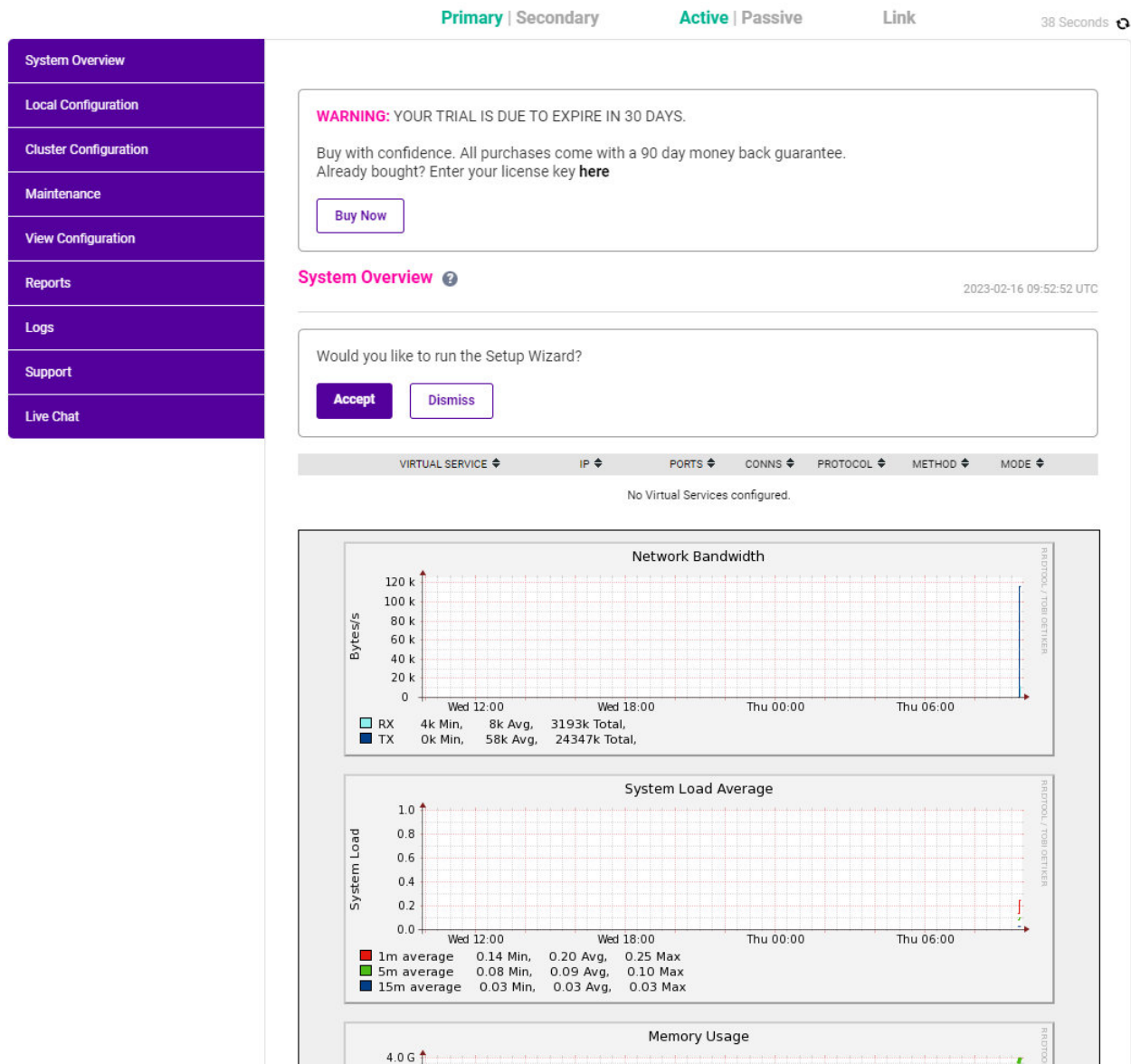
2. Log in to the WebUI using the following credentials:

Username: loadbalancer

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

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

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



3. You'll be asked if you want to run the Setup Wizard. Click **Dismiss** if you're following a guide or want to configure the appliance manually. Click **Accept** to start the Setup Wizard.



Note

The Setup Wizard can only be used to configure Layer 7 services.

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

9.4. Appliance Software Update

To ensure that the appliance(s) are running the latest software version, we recommend a software update check is performed.

Determining the Current Software Version

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

Copyright © Loadbalancer.org Inc. 2002 – 2023
ENTERPRISE VA Max - v8.9.0

English ▼

Checking for Updates using Online Update

Note

By default, the appliance periodically contacts the Loadbalancer.org update server and checks for updates. An update check can also be manually triggered as detailed below.

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

Information: Version v8.9.0 is the current release. No updates are available

4. If an update is available, you'll be presented with a list of new features, improvements, bug fixes and security related updates.
5. Click **Online Update** to start the update process.

Note

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

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

Information: Update completed successfully.

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

Using Offline Update

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





Note

Please contact support@loadbalancer.org to check if an update is available and obtain the latest offline update files.

To perform an offline update:

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

Software Update

Offline Update

The following steps will lead you through offline update.

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

Archive: No file chosen

Checksum: No file chosen

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

9.5. 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



9.6. HA Clustered Pair Configuration

Loadbalancer.org recommend that load balancer appliances are deployed in pairs for high availability. In this guide a single unit is deployed first, adding a secondary unit is covered in [Configuring HA - Adding a Secondary Appliance](#).

10. Appliance Configuration for RabbitMQ – Using Layer 4 DR Mode

10.1. Configuring the Virtual Service (VIP)

1. Using the web user interface, navigate to *Cluster Configuration > Layer 4– Virtual Services* and click on **Add a new Virtual Service**.
2. Define the *Label* for the virtual service as required, e.g. **RabbitMQ HA**.
3. Set the *Virtual Service IP Address* field to the required IP address, e.g. **192.168.87.5**.
4. Set the *Ports* field to the port that the RabbitMQ service is listening on, which by default is port **5672**.
5. Leave the *Protocol* set to **TCP**.
6. Leave the *Forwarding Method* set to **Direct Routing**.
7. Click **Update** to create the virtual service.

LAYER 4 – ADD A NEW VIRTUAL SERVICE

Label	<input type="text" value="RabbitMQ HA"/>	?	
Virtual Service	IP Address	<input type="text" value="192.168.87.5"/>	?
	Ports	<input type="text" value="5672"/>	?
Protocol	<input type="text" value="TCP"/>	▼	?
Forwarding Method	<input type="text" value="Direct Routing"/>	▼	?

10.2. Defining the Real Servers (RIPs)

1. Using the web user interface, navigate to *Cluster Configuration > Layer 4 – Real Servers* and click on **Add a new Real Server** next to the newly created VIP.
2. Define the *Label* for the real server as required, e.g. **Rabbit1**.
3. Set the *Real Server IP Address* field to the required IP address, e.g. **192.168.87.10**.
4. Click **Update**.
5. Repeat these steps to add additional RabbitMQ servers as required.

LAYER 4 ADD A NEW REAL SERVER - RABBITMQ_HA

Label	<input type="text" value="Rabbit1"/>	?
Real Server IP Address	<input type="text" value="192.168.87.10"/>	?
Weight	<input type="text" value="100"/>	?
Minimum Connections	<input type="text" value="0"/>	?
Maximum Connections	<input type="text" value="0"/>	?

11. Appliance Configuration for RabbitMQ – Using Layer 7 SNAT Mode

11.1. Configuring the Virtual Service (VIP)

1. Using the web user interface, navigate to *Cluster Configuration > Layer 7 – Virtual Services* and click on **Add a new Virtual Service**.
2. Define the *Label* for the virtual service as required, e.g. **RabbitMQ HA**.
3. Set the *Virtual Service IP Address* field to the required IP address, e.g. **192.168.2.5**.
4. Set the *Ports* field to the port that the RabbitMQ service is listening on, which by default is port **5672**.
5. Set the *Layer 7 Protocol* to **TCP Mode**.
6. Click **Update** to create the virtual service.

Layer 7 - Add a new Virtual Service

Virtual Service		[Advanced +]
Label	<input type="text" value="RabbitMQ HA"/>	?
IP Address	<input type="text" value="192.168.2.5"/>	?
Ports	<input type="text" value="5672"/>	?
Protocol		
Layer 7 Protocol	<input type="text" value="TCP Mode"/>	?

11.2. Defining the Real Servers (RIPs)

1. Using the web user interface, navigate to *Cluster Configuration > Layer 7 – Real Servers* and click on **Add a new Real Server** next to the newly created VIP.



2. Define the **Label** for the real server as required, e.g. **Rabbit1**.
3. Set the **Real Server IP Address** field to the required IP address, e.g. **192.168.2.10**.
4. Leave the **Real Server Port** field blank.
5. Click **Update**.
6. Repeat these steps to add additional RabbitMQ servers as required.

Layer 7 Add a new Real Server - RabbitMQ_HA

Label	<input type="text" value="Rabbit1"/>	?
Real Server IP Address	<input type="text" value="192.168.2.10"/>	?
Real Server Port	<input type="text"/>	?
Re-Encrypt to Backend	<input type="checkbox"/>	?
Weight	<input type="text" value="100"/>	?

Cancel
Update

11.3. Finalizing the Configuration

To apply the new settings, HAProxy must be reloaded. This can be done using the button in the "Commit changes" box at the top of the screen or by using the **Restart Services** menu option:

1. Using the WebUI, navigate to: **Maintenance > Restart Services**.
2. Click **Reload HAProxy**.

12. Testing & Verification





Note

For additional guidance on diagnosing and resolving any issues you may have, please also refer to [Diagnostics & Troubleshooting](#).

12.1. Using System Overview

The System Overview can be viewed in the WebUI. It shows a graphical view of all VIPs & RIPs (i.e. the RabbitMQ Nodes) and shows the state/health of each server as well as the state of the cluster as a whole.

This first example, shown below, shows a layer 4 DR mode VIP where all RabbitMQ nodes are healthy and available to accept connections.

	VIRTUAL SERVICE ↕	IP ↕	PORTS ↕	CONNS ↕	PROTOCOL ↕	METHOD ↕	MODE ↕	
↑	RabbitMQ_HA	192.168.87.5	5672	0	TCP	Layer 4	DR	
	REAL SERVER	IP	PORTS	WEIGHT	CONNS			
↑	Rabbit1	192.168.87.10	5672	100	0	Drain	Halt	
↑	Rabbit2	192.168.87.11	5672	100	0	Drain	Halt	
↑	Rabbit3	192.168.87.12	5672	100	0	Drain	Halt	

This second example, shown below, shows a layer 7 SNAT mode VIP where all RabbitMQ nodes are healthy and available to accept connections.

13. Technical Support

For more details about configuring the appliance and assistance with designing your deployment please don't hesitate to contact the support team using the following email address: support@loadbalancer.org.

14. Further Documentation

For additional information, please refer to the [Administration Manual](#).

15. Appendix

15.1. 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. By default the agent listens on TCP port 3333 but this can be changed if required.

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 $(92/100 * \text{requested_weight})$ to find the new optimized weight.

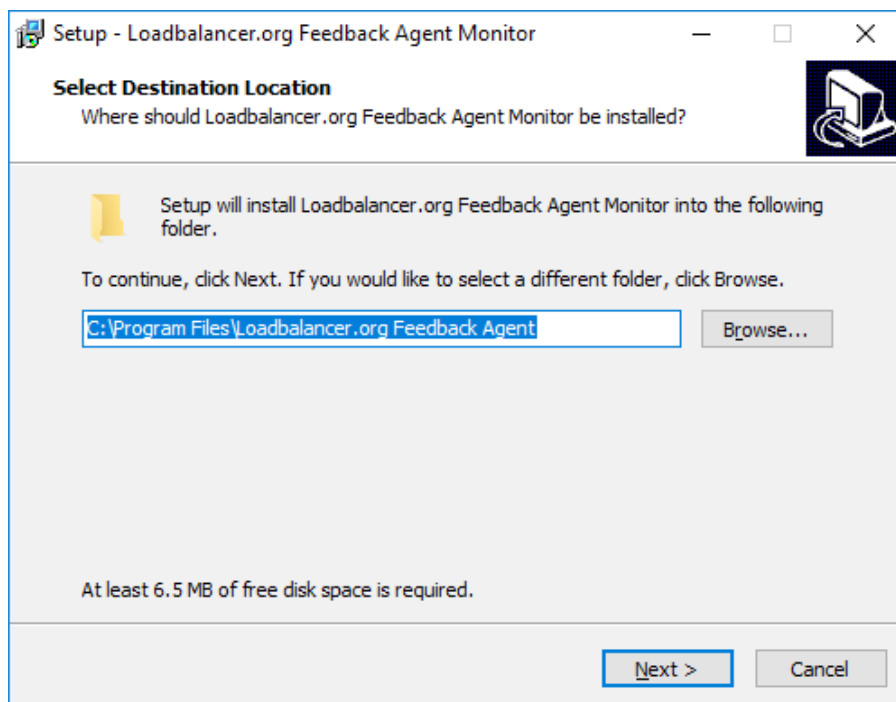
Note

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

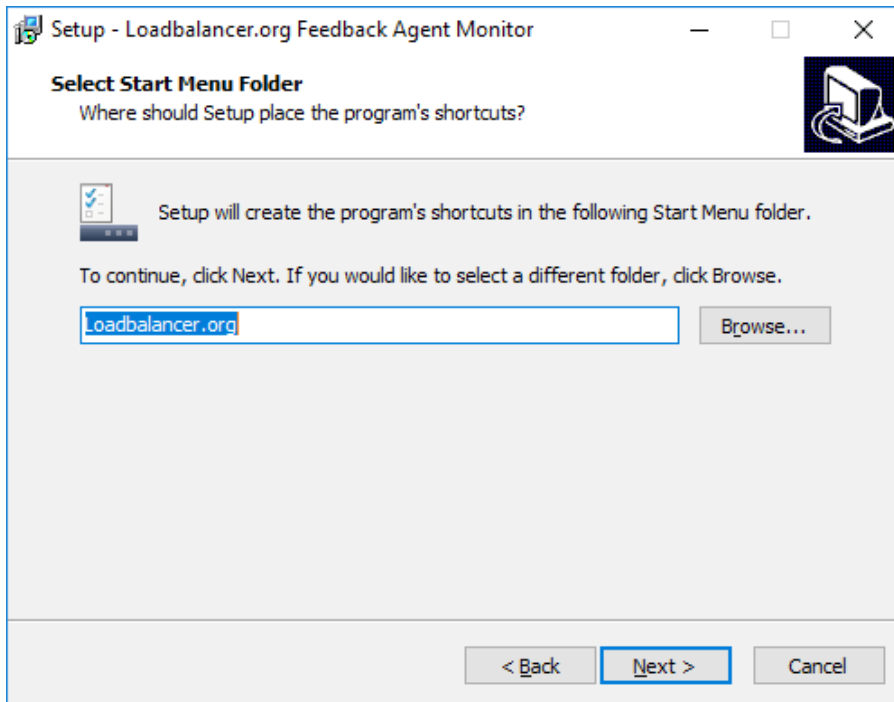
For more information about the feedback agent, please refer to [this blog](#).

Windows Agent

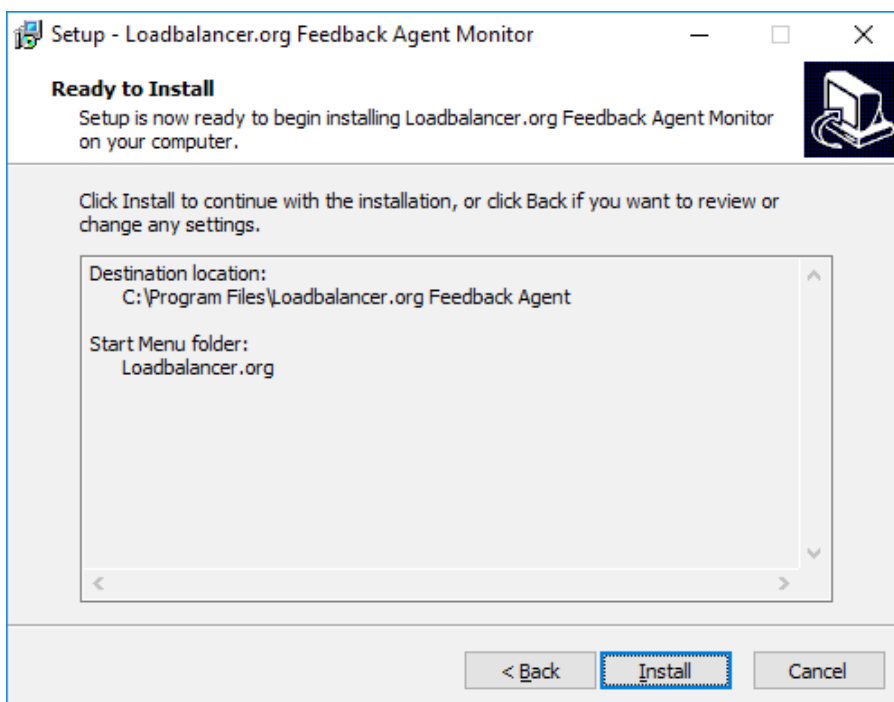
The latest Windows feedback agent can be downloaded from [here](#). To install the agent, run **loadbalanceragent.msi** on each Real Server:



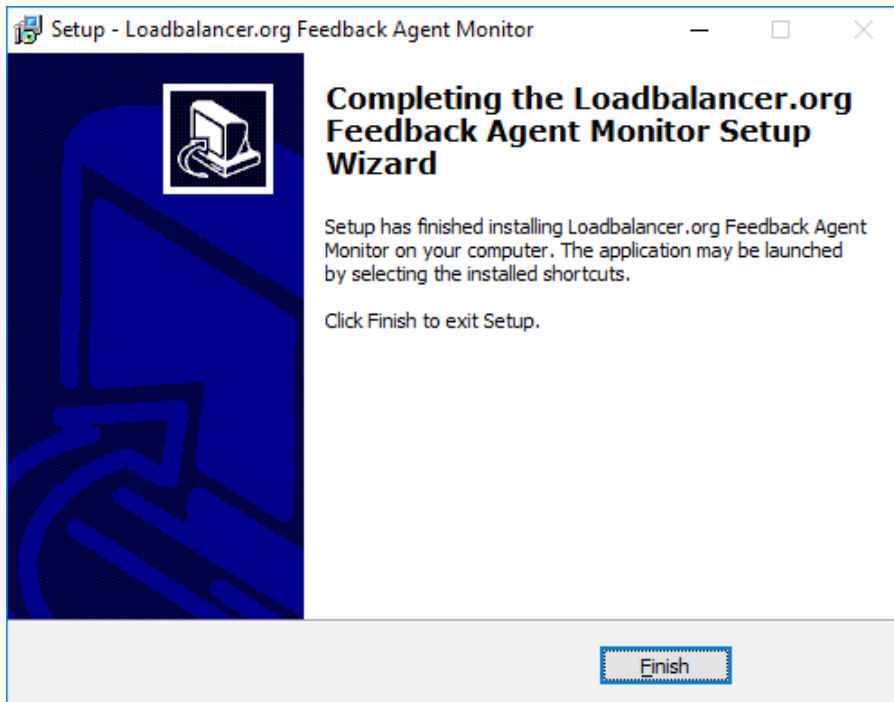
Leave the default location or change according to your requirements, click **Next**.



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



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

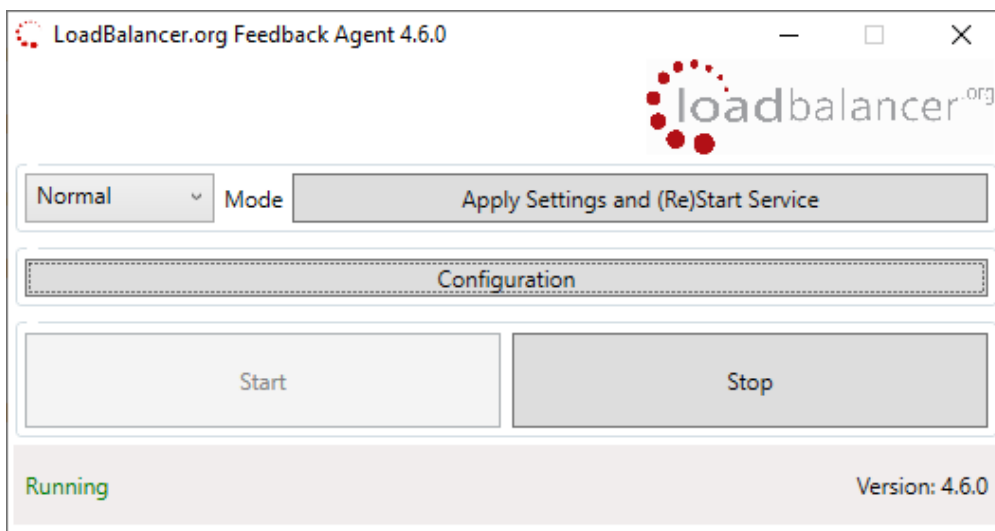


Click **Finish**.

Once the installation is complete, the Feedback Agent service is started.

Controlling the Agent

The Feedback Agent service (**LBCPUMon**) can be controlled & configured using the *Feedback Agent Service Monitor* program. By default this can be accessed from: **Start> Loadbalancer.org**.



Linux/Unix Agent

The Linux feedback agent files can be downloaded using the following links:

readme file: <https://downloads.loadbalancer.org/agent/linux/v4.1/readme.txt>

xinetd file: <https://downloads.loadbalancer.org/agent/linux/v4.1/lb-feedback>

feedback script: <https://downloads.loadbalancer.org/agent/linux/v4.1/lb-feedback.sh>



Installation & Testing

Install xinetd - if not already installed:

```
apt-get install xinetd
```

Insert 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 the agent:

```
telnet 127.0.0.1 3333
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
95%
Connection closed by foreign host.
```



Note

The agent files must be installed on each Real Server.

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 application's requirements.

15.2. Configuring HA - Adding a Secondary Appliance

Our recommended configuration is to use a clustered HA pair of load balancers to provide a highly available and resilient load balancing solution.

We recommend that the Primary appliance is configured first and 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.



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.

The clustered HA pair uses Heartbeat to determine the state of the other appliance. Should the active device (normally the Primary) suffer a failure, the passive device (normally the Secondary) will take over.

Non-Replicated Settings

A number of settings are not replicated as part of the Primary/Secondary pairing process and therefore must be manually configured on the Secondary appliance. These 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

Important

Make sure that if these settings/updates have been configured on the Primary appliance, they're also configured on the Secondary appliance.


Adding a Secondary Appliance - Create an HA Clustered Pair

Note

If you have already run the firewall lockdown wizard on either appliance, you'll need to ensure that it is temporarily disabled on both appliances whilst performing the pairing process.

1. Deploy a second appliance that will be the Secondary and configure initial network settings.
2. Using the WebUI on the Primary appliance, navigate to: **Cluster Configuration > High-Availability Configuration**.

Create a Clustered Pair

 **LOADBALANCER**

Local IP address

192.168.110.40

IP address of new peer

192.168.110.41


Password for *loadbalancer* user on peer

••••••••••

Add new node

3. Specify the IP address and the *loadbalancer* user's password for the Secondary (peer) appliance as shown in the example above.
4. Click **Add new node**.
5. The pairing process now commences as shown below:


Create a Clustered Pair

 **LOADBALANCER**

Primary

IP: 192.168.110.40

Attempting to pair..

 **LOADBALANCER**

Secondary

IP: 192.168.110.41

Local IP address

192.168.110.40

IP address of new peer

192.168.110.41


Password for *loadbalancer* user on peer

••••••••••

configuring


6. Once complete, the following will be displayed on the Primary appliance:

High Availability Configuration - primary

 **LOADBALANCER**

Primary

IP: 192.168.110.40

 **LOADBALANCER**

Secondary

IP: 192.168.110.41

Break Clustered Pair**Make Active**

7. To finalize the configuration, restart heartbeat and any other services as prompted in the "Commit changes" message box at the top of the screen.

Note

Clicking the **Restart Heartbeat** button on the Primary appliance will also automatically restart heartbeat on the Secondary appliance.

Note

For more details on configuring HA with 2 appliances, please refer to [Appliance Clustering for HA](#).

Note

For details on testing and verifying HA, please refer to [Clustered Pair Diagnostics](#).

15.3. Solving the ARP Problem

Solving the ARP Problem for Linux

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

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

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

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

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

Method 1: ARP Behaviour and Loopback Interface Changes

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

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

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

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



Step 1 of 4: Re-configuring ARP behaviour

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

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

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

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

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

Note

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

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

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

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

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

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

Note

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

Step 3 of 4: Applying the new settings

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

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

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

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

Note

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

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

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

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

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

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

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

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

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

```
ip addr ls
```

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

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

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

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

Method 2: NAT "redirect" via iptables

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

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

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

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

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

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

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

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

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

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

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

Note

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

Method 3: NAT "redirect" via nftables

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

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

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

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

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

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

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

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

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

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

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



A legacy-friendly setup may look like the following:

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

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

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

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

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

Note

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

Method 4: NAT "redirect" via firewall-cmd

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

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

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

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

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

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

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

Note

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

Windows Server 2012 & Later

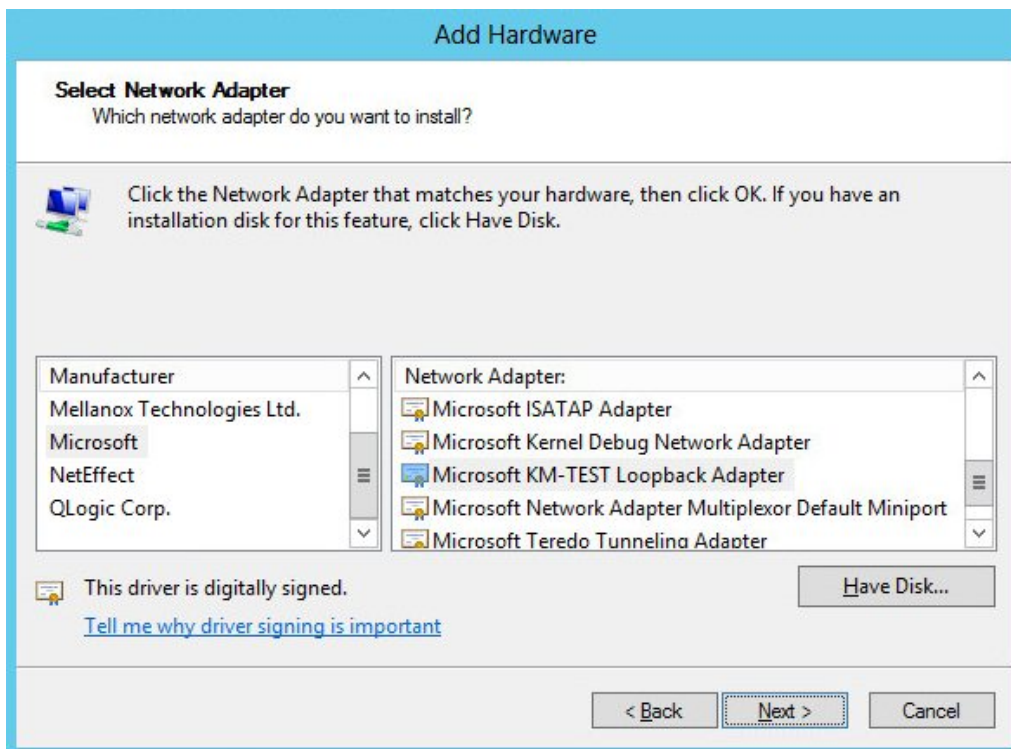
Windows Server 2012 and later support Direct Routing (DR) mode through the use of the Microsoft Loopback Adapter that must be installed and configured on each load balanced (Real) Server. The IP address configured on the Loopback Adapter must be the same as the Virtual Service (VIP) address. This enables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

In addition, steps must be taken to set the strong/weak host behavior on each Real Server. This is used to either prevent or allow interfaces to receive packets destined for a different interface on the same server.

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

Step 1 of 3: Install the Microsoft Loopback Adapter

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



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

Step 2 of 3: Configure the Loopback Adapter

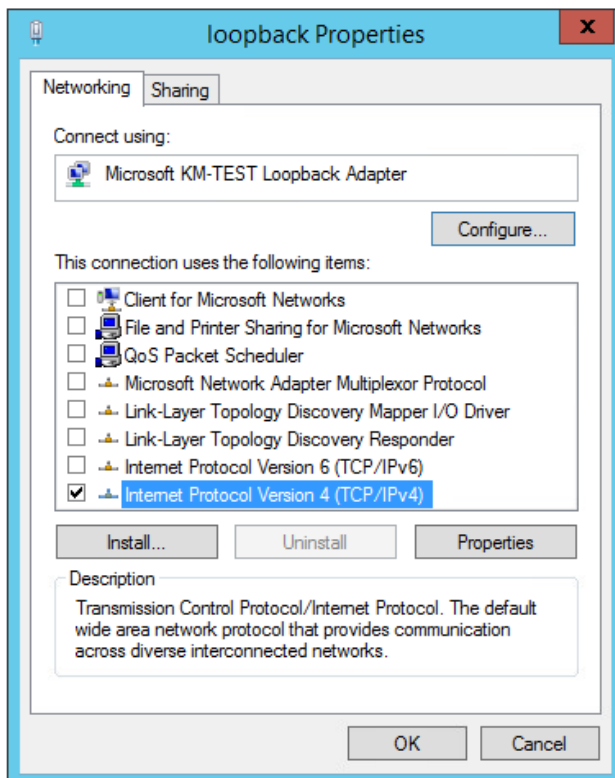
1. Open Control Panel and click **Network and Sharing Center**.
2. Click **Change adapter settings**.
3. Right-click the new Loopback Adapter and select **Properties**.

Note

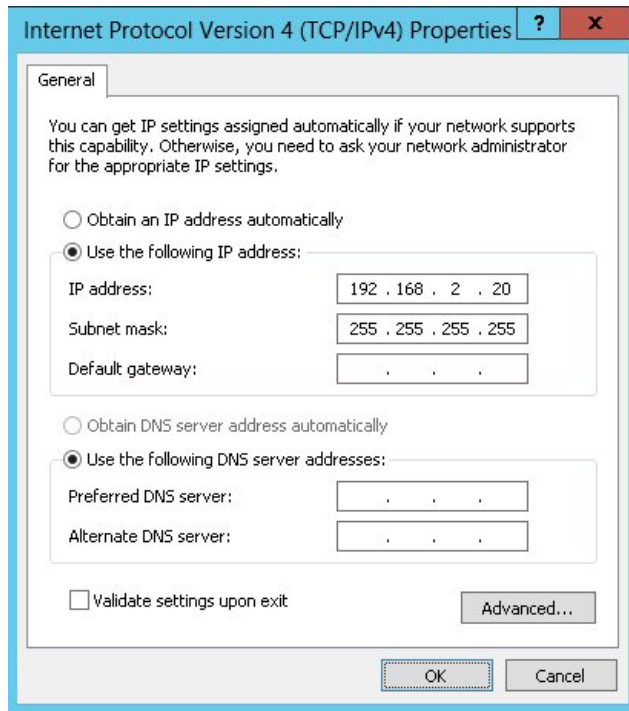
You can configure IPv4 or IPv6 addresses or both depending on your requirements.


IPv4 Addresses


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



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



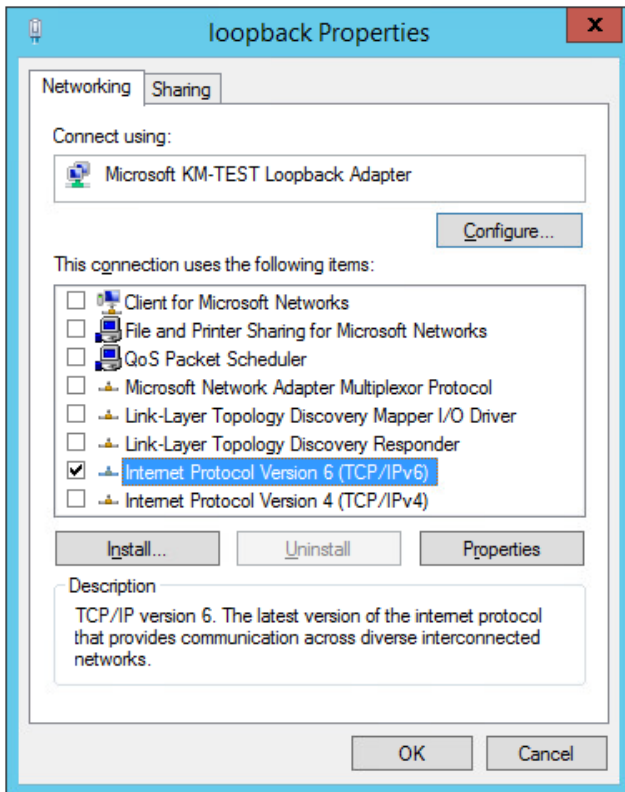
 **Note** **192.168.2.20** is an example, make sure you specify the correct VIP address.

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

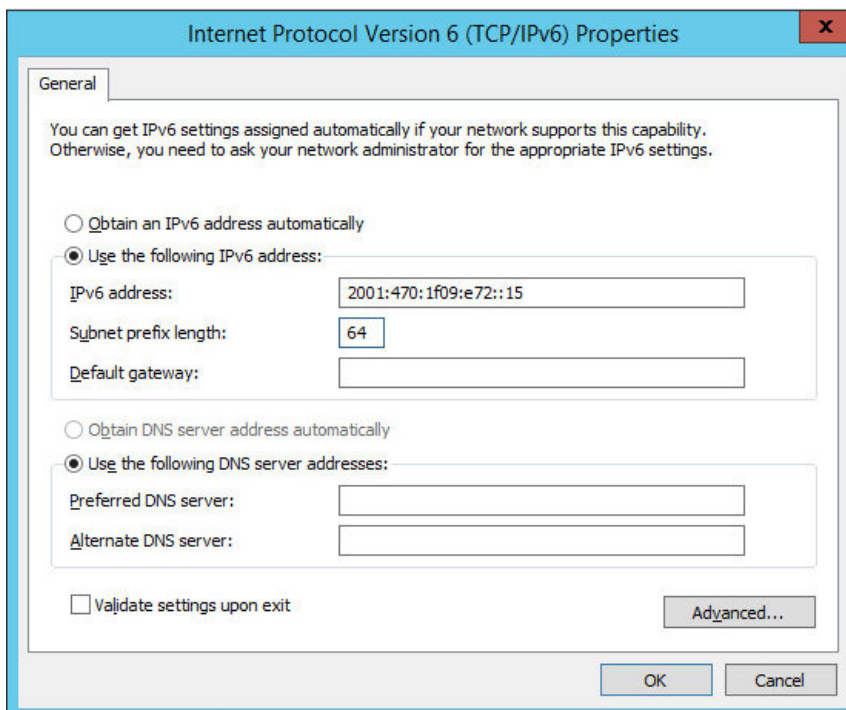
3. Click **OK** then click **Close** to save and apply the new settings.

IPv6 Addresses

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



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



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

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

added to the Loopback Adapter.

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

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

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

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

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



(!) Important

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

Option 1 - Using Network Shell (netsh) Commands

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

For IPv4 addresses:

```
netsh interface ipv4 set interface "net" weakhostreceive=enabled
netsh interface ipv4 set interface "loopback" weakhostreceive=enabled
netsh interface ipv4 set interface "loopback" weakhostsendsend=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" weakhostsendsend=enabled
netsh interface ipv6 set interface "loopback" dadtransmits=0
```

Option 2 - Using PowerShell Cmdlets

For IPv4 addresses:



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

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

For IPv6 Addresses:

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

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

16. Document Revision History

Version	Date	Change	Reason for Change	Changed By
1.0.0	29 March 2018	Initial version		AH
1.0.1	6 December 2018	Added the new "Company Contact Information" page	Required updates	AH
1.1.0	19 September 2019	Styling and layout	General styling updates	RJC
1.1.1	28 August 2020	New title page Updated Canadian contact details	Branding update Change to Canadian contact details	AH
1.2.0	1 January 2022	Converted the document to AsciiDoc	Move to new documentation system	AH, RJC, ZAC
1.2.1	28 September 2022	Updated layer 7 VIP and RIP creation screenshots	Reflect changes in the web user interface	AH
1.2.2	5 January 2023	Combined software version information into one section Added one level of section numbering Added software update instructions Added table of ports used by the appliance Reworded 'Further Documentation' section Removed references to the colour of certain UI elements	Housekeeping across all documentation	AH
1.2.3	2 February 2023	Updated screenshots	Branding update	AH
1.2.4	7 March 2023	Removed conclusion section	Updates across all documentation	AH
1.3.0	24 March 2023	New document theme Modified diagram colours	Branding update	AH



Visit us: www.loadbalancer.org

Phone us: +44 (0)330 380 1064

Phone us: +1 833 274 2566

Email us: info@loadbalancer.org

Follow us: [@loadbalancer.org](https://twitter.com/loadbalancer.org)

About Loadbalancer.org

Loadbalancer.org's mission is to ensure that its clients' businesses are never interrupted. The load balancer experts ask the right questions to get to the heart of what matters, bringing a depth of understanding to each deployment. Experience enables Loadbalancer.org engineers to design less complex, unbreakable solutions - and to provide exceptional personalized support.

