Micro Focus Security ArcSight Load Balancer

Software Version: 1.4.1

Configuration Guide

Document Release Date: January 10, 2020
Software Release Date: January 10, 2020

Legal Notices

Copyright Notice

Trademark Notices

Support

Contact Information

Phone	A list of phone numbers is available on the Technical Support Page: https://softwaresupport.softwaregrp.com/support-contact-information
Support Web Site	https://softwaresupport.softwaregrp.com/
ArcSight Product Documentation	https://community.softwaregrp.com/t5/ArcSight-Product-Documentation/ct-p/productdocs

Contents

Chapter 1 — SmartConnector Load Balancer

This guide provides information about downloading, installing and configuring the Security ArcSight SmartConnector Load Balancer application for use with event collection for ArcSight products.

Overview

Security ArcSight SmartConnector Load Balancer provides a “connector-smart” load balancing mechanism by monitoring the status and load of SmartConnectors. Currently it supports two types of event sources and SmartConnectors. One distributes the syslog input stream to syslog connectors using TLS, TCP or UDP protocol and the other downloads files from a remote server and distributes them to the file-based connectors. Note that the TLS protocol is supported for the SmartConnector for Syslog NG Daemon only.

Load Balancer ensures efficiency by distributing the load to a pool of SmartConnectors. Load Balancer supports high availability configuration with active and standby nodes. It distributes the events received to one or more SmartConnectors predefined in the SmartConnector pool.

Load Balancer is aware of the following information for SmartConnectors defined as the SmartConnector pool:

- **Availability (up or down)** – Load Balancer monitors SmartConnectors for availability. Events are not forwarded to a SmartConnector if it is not running (down). Instead, events are forwarded to the next available SmartConnector in the pool per the defined load-balancing algorithm rules.
- **SmartConnector Load** - CPU usage, memory usage, and queue drop rate for events.

Why Use Load Balancer?

Often a varying volume of events from the event source makes it difficult to configure the connector and there could be an outage in continuous event collection if any connectors go down. Load Balancer addresses these problems by distributing the events across SmartConnectors and by redistributing the events to available connectors if any connectors are down.

Load Balancer provides support to devices generating varying volumes of events, where:

- Overloaded connectors result in event loss and delayed collection
- Under-utilized connectors result in wasted resources
- Manual and tedious sizing and maintenance is necessary
- One connector becomes a single point of failure

Load Balancer is a solution for:

- Connector-smart load balancing
- Load balancing for TCP protocol without keeping the sessions sticky
- Load balancing for files
- An aggregation-preferred routing policy, which sends events from a single device to the same connector up to a certain threshold

Load Balancer together with the SmartConnector pool provides availability, reliability, and scalability.

- Load Balancer supports High Availability (HA). If the active Load Balancer node is down, a passive Load Balancer node becomes an active node and continuously collects the events.
- If a SmartConnector is down, Load Balancer forwards the events to the next available connector in the SmartConnector pool per the load balancing rules.

Load Balancer Features

Load Balancer supports the following:

- High availability (HA) mode, which can be configured with two hosts.
- Syslog type of input stream or batch files on FTP server.
- Syslog-based and file-based SmartConnectors as destinations.
- TLS, TCP and UDP protocol for syslog-type input or connectors.

Note: TLS is supported on the SmartConnector for Syslog NG Daemon only.

- Three routing policies — round robin, weighted round robin, and aggregation preferred.
- Event batching (TCP only) for better aggregation at the destination connector and better network throughput.
- Email notification for up/down status on member hosts and destination connectors.
- Load and health monitoring of connector destinations.
- Load Balancer runs either as a service or standalone application.
- TLS encryption is supported between devices and Load Balancer, Load Balancer and SmartConnectors, or both.

How Load Balancer Works

Load Balancer supports syslog-based load balancing, file-based load balancing, and several types of routing rules.

Syslog-based Load Balancing

Support for Single-line Events

The load balancer parses the input stream into a line but not to an event. It supports single-line event stream but not multi-lined events.

For TLS

For TLS, you must use a SmartConnector for Syslog NG with TLS enabled. TLS is supported over TCP syslog connections. In the destination definition of the lbConfig.xml file, change the protocol from tcp to tls. These are configurable per destination (listener).

Using TLS, incoming events will be processed automatically, as long as a self-signed certificate is imported into any devices sending events to the Load Balancer. Also, you must set up CA-signed certificates if you want to use HA; otherwise, you will have to import the certificate for both Load Balancers into all of the devices.

Using CA Signed Certificates

Load Balancer uses several digital, public-key certificates as part of establishing secure TLS communications. During the initial configuration of Load Balancer, these certificates are self-signed. In some circumstances, it might be necessary to obtain certificates digitally signed by a certificate authority (CA).

You can replace the self-signed certificate with a certificate signed by a well-known CA, such as VeriSign, Thawte, or Entrust. You can also replace the self-signed certificate with a certificate digitally signed by a less common CA, such as a CA within your company or organization.

Note: There are many well-known CAs and identifying the commonly used CAs varies with country.

Configuring TLS Certificates

This section provides instructions about configuring TLS certificates to get them digitally signed by a CA.

Before configuring the TLS Certificates, add the following global parameters in the **lbConfig.xml** file to select the certificate and keystore.

```
<globalParameters>
<properties>
<property key="ssl.cert.file" value="LBTLS.cer"/>
<property key="ssl.keystore.file" value="LB"/>
</properties>
</globalParameters>
```

Configuring the TLS certificates involves the following steps:

- "Generating a Certificate Signing Request" below
- "Getting the CSR Signed by the CA" on the next page
- "Importing the Digitally Signed Certificates into Load Balancer" on the next page

Generating a Certificate Signing Request

To obtain a digitally signed certificate, you must first generate a certificate signing request (CSR) that is presented to the CA. To generate one or more CSRs, perform the following steps on the Load Balancer server:

1. Log in to the Load Balancer server as the *root* user.
2. Create JKS Keystore and Keypair using the following command:

   ```
   /root/ArcSightSCLoadBalancer/current/jre/bin/#./keytool -keystore
   /root/ArcSightSCLoadBalancer/current/user/loadbalancer/lbcert.jks -
   storepass changeit -genkeypair -alias mykeyX -keysize 2048 -keyalg RSA
   ```

   The above command creates the **lbcert.jks** file. Enter the certificate subject information and then press Enter to use the same password used for the keystore password.

3. Generate the CSR using the following command:

   ```
   /root/ArcSightSCLoadBalancer/current/jre/bin/#./keytool -keystore
   /root/ArcSightSCLoadBalancer/current/user/loadbalancer/lbcert.jks -
   storepass changeit -certreq -alias mykeyX -file
   /root/ArcSightSCLoadBalancer/current/user/loadbalancer/lbreq.csr
   ```

Getting the CSR Signed by the CA

You should get the CSR signed by the CA.

To get the CSR signed by the CA:

1. Submit the CSRs to the CA for signature.
2. Obtain the signed certificate files from the CA.

The details of how this is done depend on the CA. For more information, consult your CA.

Importing the Digitally Signed Certificates into Load Balancer

This section provides instructions about importing the digitally signed certificates into Load Balancer. Copy the files that contain the digital certificates signed by the CA to the Load Balancer server. If the files are signed by an enterprise or organizational CA rather than a well-known CA, you must copy the CA's self-signed root certificate to the Load Balancer server.

You must import the intermediate, root, and signed certificates. You can specify the desired alias names for the intermediate and root certificates. However, the signed certificate must be imported with the same alias that was used while creating a certificate pair, which is webserver.

To import the certificate files to the Load Balancer server:

1. Log in to the Load Balancer server as the *root* user.
2. Back up the **loadbalancer.cer** file present at the following location:

   ```
   /root/ArcSightSCLoadBalancer/current/user/loadbalancer
   ```

3. Import the trusted CA certificate:

   ```
   /root/ArcSightSCLoadBalancer/current/jre/bin/#./keytool -importcert -file
   /root/ArcSightSCLoadBalancer/current/user/loadbalancer/certnew.cer -
   storepass changeit -keystore
   /root/ArcSightSCLoadBalancer/current/user/loadbalancer/lbcert.jks
   ```

   The CA certificate can be downloaded from the in-house CA server.

4. Import the signed certificate:

   ```
   /root/ArcSightSCLoadBalancer/current/jre/bin/#./keytool -keystore
   /root/ArcSightSCLoadBalancer/current/user/loadbalancer/lbcert.jks -
   storepass changeit -importcert -alias mykeyX -file
   /root/ArcSightSCLoadBalancer/current/user/loadbalancer/certsign.cer
   ```

5. Convert Keystore to P12:

   ```
   /root/ArcSightSCLoadBalancer/current/jre/bin/#./keytool -importkeystore -
   srckeystore
   /root/ArcSightSCLoadBalancer/current/user/loadbalancer/lbcert.jks -
   srcstorepass changeit -deststorepass changeit -srcstoretype JKS -
   ```

```
deststoretype PKCS12 -destkeystore
/root/ArcSightSCLoadBalancer/current/user/loadbalancer/LB.p12
```

6. Restart Load Balancer.

Import the certificate files to the Syslog Daemon connector using the following command:

```
/ArcSightSmartConnectors/current/jre/bin/#./keytool -importcert -file
/tmp/certnew.cer -storepass changeit -alias mykey -keystore
/root/ArcSightSmartConnectors/current/jre/lib/security/cacerts
```

Note: The Syslog Daemon connector can now send TLS events to Load Balancer.

For TCP

When the source is a syslog-based network process and configured to use TCP protocol, the input stream is parsed into event lines and bundled into a batch. Then the batch is distributed to one of the destinations in the destination pool.

For UDP

If a routing rule is configured to use UDP protocol, event batching does not happen. Instead, each incoming event is distributed into one of the destinations configured in the routing policy.

File-based Load Balancing

Load Balancer downloads files from an FTP server and distributes them to one of the locations associated with the file connectors. It supports batch files. File-based connectors that read and process files are good candidates for this feature.

Routing Policies

Routing policies are a set of rules that define the data distribution from a source to a set of destinations. Eligible sources are syslog servers or FTP servers. A destination pool consists of one or more syslog or file connector destinations, all of the same type. Connector types cannot be mixed within a single destination pool. Routing policies define event or file distribution rules from a source to destination pool.

The routing policies supported in Load Balancer are:

- **Round Robin:** Distributes events, batches, or files to each available destination in the destination pool in round robin fashion, beginning again at the start in a circular manner. File-based load balancing supports only the Round Robin policy.

- **Weighted Round Robin:** Distributes events in a round-robin fashion, but sends more events or batches to lightly loaded destinations.
- **Aggregation Preferred:** Events from the same source are sent to the same destination until a threshold is reached. Then, it will switch the routing to another destination. This routing policy is better suited if aggregation is enabled on connector destinations where events are sent to the same destination until certain load thresholds are met.

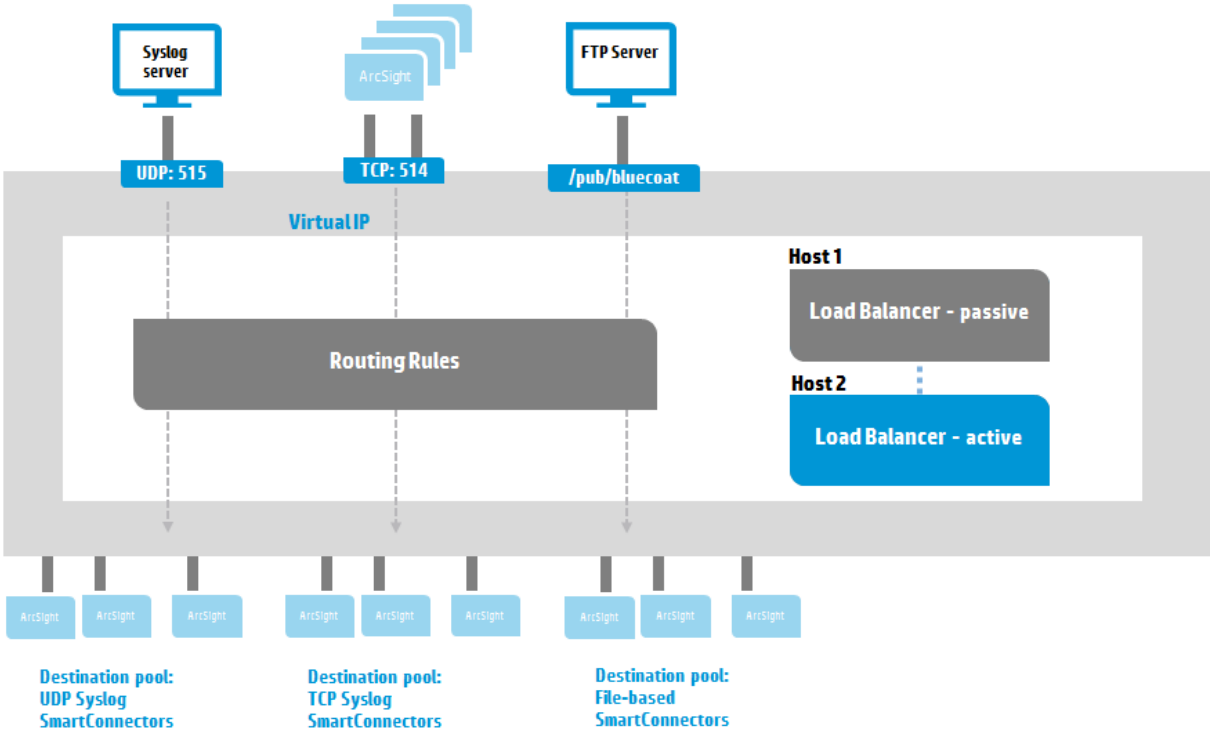
Load Balancer Modes

Load Balancer can be configured to run in three modes. To use the high availability feature, Load Balancer should be installed on two separate hosts sharing a virtual IP address. See the details at "Configuring Load Balancer in HA Mode" on page 25.

- **Standalone mode:** Load Balancer runs as a single host without supporting the high availability feature. One host with a single static IP address is required to run Load Balancer in this mode.
- **HA mode as peer:** Load Balancer runs with two hosts. The host that starts first becomes active and another host runs as passive until the first host goes down. The second host becomes active and stays active, even if the first host comes back up again.
- **HA mode as primary-secondary:** Load Balancer runs with two hosts. One host can be designated as the preferred active host. In this mode, the host marked as primary runs as active node whenever it becomes available.

Note: The High Availability feature, which is available using primary-secondary or peer mode, currently works only within the same subnet.

Load Balancer can be deployed between any syslog source, including SmartConnectors configured with CEF syslog or raw syslog destinations, or file source and SmartConnectors. The following diagram shows a Load Balancer deployment example running in HA mode. Both hosts share the common virtual IP address to handle the connection failover when an active Load Balancer host goes down. As shown in the diagram, Load Balancer can be used for three different types of input sources and destination pool types.



When configuring the routing rule, source and destination types must match. If the source is TCP syslog, the connectors in the destination pool must be TCP syslog connectors. Likewise, if the source is a file type, the connectors on the destination must be file-based connectors that expect to handle files.

When the routing rule is configured with TCP protocol, events received from the same source IP and port number are bundled into event batches. Event batching happens when any of the following conditions are met: buffer size, number of events, or batching interval. The bundled event batch, which is optional, is persisted by default on the hard drive before it is sent to the destination connector in the `${ARCSIGHT_HOME}/user/loadbalancer/lbdata/persistence/{source}` directory of the currently active node. Note that persisted event batches are not shared across the member hosts and any unprocessed event batches awaiting bundling during the shutdown are sent when Load Balancer starts up again.

Chapter 2 — Installation and Configuration

This section describes system requirements and getting started with Load Balancer, including pre-deployment requirements, Ethernet configuration, Load Balancer installation, and Load Balancer configuration.

Load Balancer is an independent component, not packaged with SmartConnectors.

System Requirements

The following section outlines the minimum system requirements for ArcSight SmartConnector Load Balancer.

General Setup

ThefollowingsectiondescribesthesoftwareandplatformrequirementsforallreleasesofMicroFocusSmartConnectorLoadBalancer.

Servers should be dedicated to load balancing (not running other applications.)

For high availability (HA), there should be two separate servers, one for the active or primary Load Balancer and another for standby or secondary Load Balancer. They will share a Virtual IP address, so they should be in the same network location.

In addition, use the standard hardware required to deploy more than one SmartConnector to create the pool of SmartConnectors. See the SmartConnector documentation for details.

Hardware Requirements

- CPU: 2 CPU X 4 Cores each (2 x Intel E5620, quad core, 2.4 Ghz or better)
- RAM: 16 GB
- Disk: 60 GB
- Number of network interfaces—1 Dedicated Gig Ethernet interface

Note: To achieve better performance, use a server with higher system specifications.

Software or Platform Requirements

- Red Hat Enterprise Linux (RHEL) 6.8, 7.2, and 7.4 (64-bit only)
- Certified: Red Hat Enterprise Linux (RHEL)7.3 (64-bit only)

- Supported: CentOS Linux 6.8, 7.2, and 7.4 (64-bit only)
- Certified: CentOS Linux 7.3 (64-bit only)

SmartConnector Requirements

- SmartConnector release 7.1.4.7475 or later
- Syslog daemon, Syslog NG daemon, and/or file-based SmartConnector

Downloading Load Balancer

Download the 64-bit executable and the *Security ArcSight SmartConnector Load Balancer Configuration Guide* from the Support Web site (https://softwaresupport.softwaregrp.com/).

For a successful Load Balancer installation, follow the installation procedures documented in "Installing the Load Balancer" on page 20.

Verifying Your Files

Micro Focus provides a digital public key to enable you to verify that the signed software you received is indeed from Micro Focus and has not been manipulated in any way by a third party.

Visit the following site for information and instructions:

https://entitlement.mfgs.microfocus.com/ecommerce/efulfillment/digitalSignIn.do

Preparing for Deployment

To run in HA mode, the following pre-requisites must be met:

- Have two hosts with static IP addresses to install Load Balancer.
- Have a single, unused address for the VIP to run Load Balancer in HA mode.
- Create an Ethernet configuration file to support failover migration.

Configuring the Ethernet Connection

Begin with creating an Ethernet configuration file to support failover migration.

To configure the Ethernet connection:

1. Before installation, identify the machine or machines where Load Balancer will be installed. To enable HA, two machines and one virtual IP address are needed (all in the same subnet).

2. If Load Balancer is run by a non-root user, be sure to give sudo capability to the user. For example, if `arcsight` is the user that installs and runs Load Balancer, add the `arcsight` user and add `sudoer` capability with `NOPASSWD`. See the following example:

   Note: Ignore this step if Load Balancer is installed as root.

```
# adduser arcsight   // Creates arcsight group and adds the user to the
group.
# sudo visudo

// Add the following line, and exit.
  arcsight        ALL=(ALL)       NOPASSWD:ALL
```

   Note: Steps 3 and 4 can be skipped if Load Balancer is deployed in standalone mode. Note that step 3 and 4 can vary depending on the OS version and flavor. Use the instructions as a reference. Get help from the network administrator to execute the steps below.

3. When using two machines for HA, create a network profile or Ethernet configuration file on each machine. In the supported distributions of Linux, this file is usually located in the `/etc/sysconfig/network-scripts` directory.
   a. Go to the directory and verify that the file has the primary network interface (usually 'eth0') configuration. The `IPADDR` value of this file should show the IP address assigned to this machine. A similar configuration file needs to be created for the virtual IP address.
   b. Log on as a privileged administrator and go to the directory where the Ethernet profiles are located.

```
# cd /etc/sysconfig/network-scripts
```

   c. Copy the default `eth0` configuration to `eth0:1`.

```
# cp ifcfg-eth0 ifcfg-eth0:1
```

   d. Edit `ifcfg-eth0:1` to modify `DEVICE` to `eth0:1` and `IPADDR` to a virtual IP address and save the file.

   Note: `ONBOOT` must be set to `no` in order to prevent the VIP address from being bound to the host automatically upon system reboot. Otherwise, the virtual IP address needs to be released manually when another host is running as the active node or it will lose the connection from the source devices.

```
DEVICE=eth0:1
IPADDR=<virtual-ip-address> # for example, 192.168.1.255
ONBOOT=no
NM_CONTROLLED=no
```

```
ARPCHECK=no
BOOTPROTO=static
```

4. Verify the full path of the `ifup` command, usually `/sbin/ifup`. Make note of the full path of the `ifup` command and Ethernet profile.

Installing the Load Balancer

Before beginning your installation, obtain the Load Balancer binary (see "Downloading Load Balancer" on page 18) and configure the Ethernet connection (see "Configuring the Ethernet Connection" on page 18), if needed. If Load Balancer will be running in HA mode, install Load Balancer on each host.

The installer runs both in console mode and GUI mode. Follow the instructions in one of the following sections for the appropriate mode:

- "Installing Load Balancer in Console Mode" below
- "Installing the Load Balancer in GUI Mode " on page 23

Installing Load Balancer in Console Mode

To install the Load Balancer files in console mode:

1. Obtain the Load Balancer binary file and copy it to the desired location.
2. Run the installer.

> **Note:** The '`-i console`' mode is automatically selected by default if you do not use graphical display or if the DISPLAY variable is not set. It can also be specifically invoked using the `-i console` switch as shown here.

```
# sh ArcSightSCLoadBalancer-<build-number>.bin -i console

Preparing to install...

Extracting the JRE from the installer archive...

Unpacking the JRE...

Extracting the installation resources from the installer archive...

Configuring the installer for this system's environment...

Launching installer...


Graphical installers are not supported by the VM. The console mode will be
used instead...
```

```
===============================================================================
===

ArcSight SmartConnector Load Balancer         (created with
InstallAnywhere)

-------------------------------------------------------------------------------
---

Preparing CONSOLE Mode Installation...

===============================================================================
===

Introduction

------------

The ArcSight installer guides you through the installation of the ArcSight

SmartConnector Load Balancer.


ArcSight recommends that you quit all other programs before continuing
with

this installation.


Click the ‘Next’ button to proceed to the next window. If you want to
change something on a previous window, click the ‘Previous’ button. To
cancel this installation at any time, click the ‘Cancel’ button.


PRESS <ENTER> TO CONTINUE:
```

3. Type an absolute path or just press **Enter** to accept the default location.

```
Choose Install Folder

---------------------

Select an installation folder. When upgrading from a previous version,
select the folder that contains the currently installed ArcSight
SmartConnector Load Balancer

Where to install:

  Default Install Folder: /home/arcsight/ArcSightSCLoadBalancer
```

```
ENTER AN ABSOLUTE PATH, OR PRESS <ENTER> TO ACCEPT THE DEFAULT

      :
```

4. Type the option number that corresponds with the shortcut or link to be created for Load Balancer, if any. Press **Enter**.

```
Choose Link Location

Where would you like to create links?

  ->1- Default: /home/arcsight

    2- In your home folder

    3- Choose another location...

    4- Don't create links

ENTER THE NUMBER OF AN OPTION ABOVE, OR PRESS <ENTER> TO ACCEPT THE
DEFAULT

      :
```

5. Check the pre-installation summary before proceeding to the installation, then press **Enter** to start the installation.

```
Pre-Installation Summary

------------------------

Review the following information before continuing:

Product Name:

    ArcSight SmartConnector Load Balancer

Install Folder:

    /home/arcsight/ArcSightSCLoadBalancer

Link Folder:

    /home/arcsight

Install Set:

    Typical

PRESS <ENTER> TO CONTINUE:
```

6. Upon completion, the screen displays the location where Load Balancer is installed.

```
Installing...
```

```
-------------
 [=================|=================|=================|=================]
 [-----------------|-----------------|-----------------|-----------------]
===============================================================================
Installation Complete
---------------------
The core components of the ArcSight SmartConnector Load Balancer have been
successfully installed to:
   /home/arcsight/ArcSightSCLoadBalancer
PRESS <ENTER> TO EXIT THE INSTALLER:
```

7. Press **Enter** to exit.

Installing the Load Balancer in GUI Mode

To install the Load Balancer files in GUI mode:

1. Copy the Load Balancer binary file to the desired location.
2. Run the installer.

   ```
   # sh ArcSightSCLoadBalancer-<build-number>.bin
   ```

   The installer loads and the Introduction screen displays. Click **Next**.
3. Accept the default path or enter a new path in the "Where to install" field. Click **Next**.

   Note: If you are logged in as `root`, the install path is `/root/ArcSightSCLoadBalancer`.
4. Select your preferred option for creating a link to Load Balancer (the link folder). Click **Next**.
5. Review installation information in the Pre-Installation Summary. Click **Previous** to make changes or click **Install** to install Load Balancer.
6. Review the installation location and click **Done** to quit the installer.

Uninstalling Load Balancer

Load Balancer can be uninstalled using the GUI or console mode.

To uninstall Load Balancer in GUI mode:

1. Run the `./Uninstall_ArcSightSmartConnectorLB`script in the `$ARCSIGHT_HOME/current/UninstallerData` folder to launch the GUI uninstaller. For example:

```
./Uninstall_ArcSightSmartConnectorLB -i swing
```

2. Follow the screen prompts.

To uninstall Load Balancer in console mode:

1. Run the ./`Uninstall_ArcSightSmartConnectorLB -console` command to uninstall Load Balancer in console mode. For example:

```
./Uninstall_ArcSightSmartConnectorLB -i console
```

2. Add `'-i silent'` to launch the silent mode installation.

Configuring Load Balancer

Load Balancer can be configured for standalone mode or high availability. See the appropriate section:

- "Configuring Load Balancer in Standalone Mode" below
- "Configuring Load Balancer in HA Mode" on the next page

Configuring Load Balancer in Standalone Mode

Only one host is needed when Load Balancer runs in standalone mode.

To configure Load Balancer in standalone mode:

1. Log on to the host.
2. Go to `$ARCSIGHT_HOME/config/loadbalancer`. Copy the `lbConfig.xml.template.standalone` file to the `$ARCSIGHT_HOME/user/loadbalancer/` directory to configure Load Balancer to run in standalone mode.
3. Go to `$ARCSIGHT_HOME/user/loadbalancer` directory and rename the file `lbConfig.xml`.
4. In the `lbConfig.xml` file, configure the host under the `memberHost` parameter.
   a. List the host for Load Balancer.
   b. Match "memberIdentity " on page 28 and the `memberHosts/memberHost` name so that Load Balancer can identify itself. See "Configuring MemberHosts in Standalone Mode" on page 39 for more information.
   c. Set `isPrimary=true`.
5. Configure the routing and other parameters. Routing rules must be defined to have the events distributed from a source to a set of destinations (SmartConnectors). See "Syslog Load Balancing

Routing Rule Example" on page 43 or "File Load Balancing Routing Rule Example" on page 47 for more information.

a. Configure destinations and destination pools.

b. Configure sources.

c. Configure routing rules.

6. Configure the web server.

> **Note:** The web server configuration settings must be configured. The funtion will be enabled in a future release for remote management. The value can be changed in the future, but a value must be entered to proceed.

7. Finish configuration. Refer to the Configuration Parameters and the Sample Configuration File for more information. Other optional configuration settings include:
   - Notification
   - Statistics Logging
8. Start the destination connectors before you start Load Balancer to ensure that Load Balancer can query the destinations for connector health and load.
9. Go to the `$ARCSIGHT_HOME` directory on the host and start Load Balancer.

```
# bin/arcsight loadbalancer
```

> **Note:** If there are any configuration errors, Load Balancer will not start. Instead, it logs the configuration error messages at `logs/loadbalancer.log`. If this happens, fix the issue associated with the error message and start Load Balancer again.

Configuring Load Balancer in HA Mode

If Load Balancer will be run in High Availability (HA) mode, first decide the type (peer or primary-secondary HA) and configure Ethernet file accordingly. (See "Preparing for Deployment" on page 18.) Configure either the HA primary member host or the one that will be started first in peer mode with the full configuration in the XML configuration file. The secondary host (for primary-secondary configuration) or the passive host (for peer configuration) that starts second must be configured with the settings for member hosts and web service only. Other configuration settings will be synchronized when the second member host is started after the first member host.

> **Note:** The host that starts first will overwrite the configuration file of the host that is started second. If the Load Balancer host with an incomplete configuration file is started first, the configuration can be lost. It is recommended to make a backup of the completed configuration file before starting Load Balancer.

To configure Load Balancer in HA mode:

1. Log on to the primary member host or the one where the Load Balancer will start first (for peer configuration).
2. Go to `$ARCSIGHT_HOME/config/loadbalancer`. Copy the `lbConfig.xml.template` file to the `$ARCSIGHT_HOME/user/loadbalancer/` directory if Load Balancer will be running in HA mode.
3. Go to `$ARCSIGHT_HOME/user/loadbalancer` directory and rename the file `lbConfig.xml`.
4. In the `lbConfig.xml` file, configure both participating member hosts and the routing rules.
   a. List both participating member hosts for Load Balancer. (For security reasons, Load Balancer only communicates with known hosts using configured ports.)
   b. Match up `memberIdentity` with one of `memberHosts/memberHost/name` so that Load Balancer can identify itself. See "Configuring MemberHosts as Peer" on page 40 and "Configuring MemberHosts as Primary-Secondary" on page 41 for more information.
   c. For primary-secondary configuration, set `isPrimary=true` on the primary host. On the secondary host, set `isPrimary=false`.
   d. For peer mode, set both hosts to `isPrimary=false`.
   e. Refer to "Configuration Parameters" on the next page to configure a secondary host and finish the host configuration.
5. Configure the routing and other parameters. Routing rules must be defined to have the events from a source be distributed to a set of destinations (SmartConnectors). See "Syslog Load Balancing Routing Rule Example" on page 43 or "File Load Balancing Routing Rule Example" on page 47 for more information.
   a. Configure destinations and destination pools.
   b. Configure sources.
   c. Configure routing rules.
6. Configure the web server.

   Note: The web server configuration settings must be configured. The funtion will be enabled in a future release for remote management. The value can be changed in the future, but a value must be entered to proceed.

7. Finish optional configuration. Refer to the Configuration Parameters and the Sample Configuration File for more information. Other optional configuration settings include:
   - Notification
   - Statistics Logging
8. Log on to the second host and do the following:

a. Go to `$ARCSIGHT_HOME/config/loadbalancer` and copy the `lbConfig.xml.template` file to the `$ARCSIGHT_HOME/user/loadbalancer/` directory. Go to `$ARCSIGHT_HOME/user/loadbalancer` directory and rename the file `lbConfig.xml`.

b. Edit member hosts in the `lbConfig.xml` file. Make sure that `memberIdentity` is different from the one configured following step 4b.

c. The routing rule can be copied from the configuration file created in step 5 if preferred, but it is not required because the configuration values will be synchronized and persist after startup.

Note: If a firewall is enabled, Load Balancer member hosts may not be able to discover each other. The configured port in `memberHost` must be open. Also, the `webserver` needs to be configured to start Load Balancer.

9. Configure the firewall to open the ports on both hosts as needed to allow the two participating hosts to detect each other. Configure the firewall rule as needed. The configured ports include: `memberHosts/vipPingPort, memberHost/port` and all listening ports configured in `source` and outbound ports configured for `destinations`.
10. Start the destination connectors before you start Load Balancer to ensure that Load Balancer can query the destination for connector health and load.
11. Go to the `$ARCSIGHT_HOME` directory on the primary or active host and start Load Balancer.

```
# bin/arcsight loadbalancer
```

Note: The virtual IP address is obtained by the host where Load Balancer is first started when Load Balancer is configured as peer. If Load Balancer is configured as primary-secondary, the virtual IP address will be used by the primary host.

Note: If there are any configuration errors, Load Balancer will not start. Instead, it logs the configuration error messages at `logs/loadbalancer.log`. If this happens, fix the issue associated with the error message and start Load Balancer again.

12. Log on to the secondary or passive host and start Load Balancer.

Configuration Parameters

Configure Load Balancer with the following parameters. They fall into several basic categories:

- "memberIdentity " on the next page
- "memberHosts" on the next page
- "notification " on page 29
- "routing" on page 30

- "statisticsLogging" on page 34
- "webServer" on page 34
- "globalParameters" on page 35
- "clusterconfigurations" on page 38

memberIdentity

memberHost in the memberHosts section defines the list of hosts that run Load Balancer, where each member host must have a unique name. The value of memberIdentity should be configured with the member host name that identifies the current host.

Verify that:

- The valid name consists of alphanumeric characters without spaces.
- The matching names are found in memberHosts/memberHost/name.
- The host configuration is the configuration for the current node.

memberHosts

Configure the list of member hosts that participate in load balancing in this section. The Load Balancer mode is determined by this configuration. Up to two member hosts are allowed.

- vipAddress: Specifies the virtual IP address when running Load Balancer with two hosts to enable HA mode.
- vipPingPort: Specifies the port used internally to detect the virtual IP binding status. Change the value if this port is being used by another application. (Default port is 9090.)
- memberHost: Configures the participating host where Load Balancer will be installed and running.
  - name: Specifies a unique name that identifies the host.
  - address: Specifies the IP address of the participating host. Load Balancer must be installed on this host.
  - port: Specifies the port number used by the underlying library for HA support.
  - isPrimary: Specifies the running mode for Load Balancer.
    - Set this value to true to designate a primary host when Load Balancer is running in primary-secondary mode.
    - Only one host can be configured as the designated primary host.
    - To run Load Balancers in peer mode, set this value to false for both member hosts.
  - vipBindCommand: Specifies the full command used to bind the virtual IP address to this host. Prior to configuring this, the Ethernet connection virtual IP address should have been configured. See "Configuring the Ethernet Connection" on page 18 for details.

- In Linux, `/sbin/ifup` shows the Ethernet configuration.
- Be sure to use the absolute path when specifying the command. For example, if the virtual IP address profile is located in:

  `/etc/sysconfig/network-scripts/ifcfg-eth0:1`
  specify:

  `sudo /sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1.`

  Note: If Load Balancer is running as the root user, remove `'sudo'`.

- `vipUnbindCommand`: Specifies the full command used to unbind the virtual IP address from this host. It defines the counter command for binding. Refer to the details in the previous `vipBindCommand.`

`notification`

The configuration information provided in this section sets up notifications when certain events occur, such as when a member host goes up or down, or when a destination host goes down or up.

- `enable`: Specifies whether the sending of notification is enabled or disabled. Set this value to `true` to enable notifications.
- `enabledNotification`: Specifies the events for which notifications are sent. Notifications are supported for the following types of events:
  - `MemberHostUp`
  - `MemberHostDown`
  - `DestinationUp`
  - `DestinationDown`

  Notifications are sent only for specified supported events. For example, if only the `MemberHostDown` event is listed, the notification will be sent only when one of the configured hosts is down.

  - `event`: Specifies the events for which a notification will be sent.
    - `name:` Specifies the event name.
    - `message:` Specifies the custom message. If undefined, a default message is used.
- `email`: Configures the email sender, receiver, prefix, and SMTP server.
  - `prefix`: Configures the value used to tag the notification message in the subject line. When not configured, the subject will not have a prefix tag.
  - `recipients`: Specifies a list of one or more valid email addresses of the recipients. Separate each email address by a space.
  - `sender`: Specifies the sender's email address.
  - `smtpServer`: Specifies the SMTP server configuration. If this value is not configured, the email will not be sent.

routing

Use this section to define the routing rules. Data will be received from the source machine and distributed to the destinations in the destination pool. In routing configuration, every name should be unique whether the name is used for source or destination. The source cannot be referenced in more than one routing rule. The destination can be referenced in more than one destination pool.

When configuring a routing rule, the incoming and outgoing protocol used for one routing rule must be the same. For example, if routing rule A has source configured with TCP, destinations in the destination pool in routing rule A must be configured with the same TCP. TLS and TCP destinations can be mixed. If the source is configured with UDP, destinations in the same routing rule must be configured with UDP. Note that TLS cannot be mixed with UDP.

- `sources`: List of sources
  - `source`: Specifies the data ingress.
    - `name`: Specifies the unique name that identifies the source.
    - `type`: Specifies the source type. Valid source types are `file` and `syslog`.
      - Specify `syslog` if the events are fed from a syslog server or syslog connector.
      - Specify `file` if files are to be distributed to a destination.
    - `protocol:` Specifies the protocol that Load Balancer will use to listen:
      - For `syslog` type, `tls`, `tcp` and `udp` are supported.
      - For `file` type, `ftp` is supported.
        - `host:` If the source is configured as a `file` type, specify the host IP address, host name, or FQDN from which Load Balancer will download the files.
      - `port`: Specifies the port used:
      - For the `syslog` type, specify the port where Load Balancer will be listening. All port numbers must be unique or there will be a binding error.
      - For the `file` type, this specifies the FTP server port to which Load Balancer connects to download files. Skip the configuration of this value if FTP server is configured with the default port. (The default FTP port is 21.)

The following configuration values apply only to the `file` type source.

- `path`: Specifies the path where the files are located in the FTP server. This path should be based off the FTP root directory. For example, if the FTP root directory is configured as `/ftp/pub` and the files are located under `/ftp/pub/source`, specify `/source` to this value.
- `username`: Specifies the username used to log into the FTP server.
- `password`: Specifies the password configured for the user. The plaintext password is encrypted and persisted during the Load Balancer startup.

- `fileFilter`: Specifies the Java-style regular expression used to filter the files to download from the specified path. For example, `.*log` will filter the files with names ending with 'log'. To download the files with names starting with 'Simple' and ending with 'log', use `Simple.*log`.

  Note: When uploading files to the source FTP server, be sure to use a temporary file name and specify the filter in the `fileFilter` parameter to filter out temporary files, otherwise Load Balancer may download an incomplete file. After file upload is complete, rename the file to an original name.

- `recursive`: Specifies that Load Balancer downloads the file recursively from subdirectories when set to `true`. By default, it is set to `true`.

  Note: Load Balancer cannot handle a file larger than 1 GB – 1 byte (approximately 1GB). When Load Balancer detects a file that exceeds the maximum size, it logs an error message and continues to the next file.

- `localWorkDirectory`: Specifies the existing path to the directory where Load Balancer will place the downloaded files temporarily before they are actually sent out to the destination. Configuring this field is required when FTP is configured with the `file` source type.
- `moveToDirectory`: Specifies the directory on the source to which the files should be moved after the files are successfully delivered to the destination. This directory can be created as a sub-directory under the source directory that is specified in path or can be located in another location. In the case of moving the files to a sub-directory of the source files, be sure to provide a name that starts with dot so that it is treated as a hidden directory such as `.done`. Otherwise it will recursively download the files from sub-directories and move them to another nested sub-directory unless recursive is set to `false`. See the option to turn off recursive search. If this is set to `blank` or omitted, files are deleted instead.
- `passive`: FTP server can be configured to run in passive or active mode. Set this value to `true` if FTP server is running in passive mode. Otherwise set to `false`.

- `destinations`: List of destinations
  - `destination`: Specifies a destination where events or file will be sent.
  - Only connectors are supported as destinations. Identify the connectors to be used as destinations and configure the following values:
    - `name`: Specifies a unique name that identifies the destination.
    - `type`: Specifies the destination type. Valid destination types are `file` and `syslog`. The type must match the connector type. If a syslog connector is used as a destination, configure `type` as `syslog`. If the connector is reading files from certain directory such as a Bluecoat connector, configure type as `file`.
    - `host`: Specifies the destination address where the connector is installed. Applicable whether the `type` is set to `syslog` or `file`.

- `protocol`: Specify the protocol used to send data to the destination connector.
  - For `syslog` type, `tls`, `tcp` and `udp` are supported.
  - For `file` type, `ftp` and `scp` are supported. When `ftp` is used, the connector installation host should be running an FTP server to receive the file from Load Balancer.
- `port`: For `syslog` destinations, this value specifies the configured port where the connector listens for events. This port number should match the port number found in the `agent.properties` file of the destination connector. For `file` type, this can be skipped if the default ports are being used. The default FTP port is 21 and SCP is 22.

The following configuration values are applicable only to `file` type source.

- `username`: Specifies the username used to log into the FTP server or to run an `scp` command.
- `password`: Specifies the password set for the user. This password will also be encrypted when the Load Balancer starts up.
- `path`: Specifies the path to where the files will be moved. If protocol is configured as `ftp`, this path should be relative to the FTP root directory. For `scp`, it should be a full path.
- `knownHostsFile`: Specifies the file path of known hosts file if the destination protocol is `scp`. This file should contain the host key used for SSH connections, which is usually added to `$HOME/.ssh/known_hosts` on an initial SSH connection to a specified host. Specify the path of default known hosts file or the one created for Load Balancer testing, if it exists. Note that `$ARCSIGHT_HOME/user/loadbalancer` is assumed to be the base directory if the path does not start with "/". Currently `ssh-rsa` or `ssh-dsa` are accepted as valid algorithms.

Specify the destination connector information using in the following section to enable Load Balancer to communicate with the connector and to check the health and load. Before configuring the following values, first go to the connector installation directory on the machine where the connector is installed and ensure that remote management is enabled and get the port number configured for remote management. Corresponding property names for these two values in `agent.properties` are `remote.management.enabled` and `remote.management.listener.port`. If the destination is a file connector, the agent name— which is specified near the end of the installation wizard process during the connector configuration step and persisted into destination descriptor—will be needed. Note that agent name within the container should be unique when more than one connector is configured within the container.

- `additionalParameters/properties`
  - `remote.management.listener.port`: Specifies the value of `remote.management.listener.port` of the destination connector.
  - `load.expression`: Specifies custom load-level calculation expressions as per-destination overrides. Expressions can be used to favor certain destinations over others. Weaker

destinations can be pre-favored to be less utilized by having a large constant value added or multiplied to their load.

Note: For information on configuring an expression to calculate the load per destination, see: "Calculating Loads for Routing" on page 89.

- `agent.name`: Provide this value only if the destination is a file connector.

When you install the connector, the default port is specified under the connector installation in `config/agent/agent.defaults.properties` as `remote.management.listener.port`. If you change the `remote.management.listener.port` property value to anything other than the default, then this property is present in `agent.properties` under `user/agent`. In that case, you must update the change in the value to the correct value in the `agent.properties` file.

Note: Currently, Load Balancer only works with connectors that use default remote management user name and password values. This will be addressed in a future release.

For management of certificates from destinations, Load Balancer creates a directory on the Load Balancer machine such as `<ARCSIGHT_HOME>/certs` and populates downloaded certificates from the destinations under this folder. The automatic download and import of certificates is done in the background.

IMPORTANT:
On remote connector installations (destination connectors), turn on the remote management enabled flag. In the `user/agent/agents.properties` file, add `remote.management.enabled=true`. Do this before starting the connectors.

Start the destination connectors before starting Load Balancer. Doing so ensures Load Balancer is able to query destinations for usage, load, and health statistics. More importantly, this also lets Load Balancer contact the connector host on the provided port and download the certificates for the connectors, which then enables the destination monitoring. If the connector is not up when Load Balancer starts, Load Balancer will check periodically to see if the connector comes up and then includes it for destination monitoring.

- `destinationPools`: List of destination pools
  - `destinationPool`: Specifies the destination group that can handle the same type of events. All destinations in one destination pool must be of the same type.
    - `name`: Specifies a unique name that identifies the destination pool.
    - `destinations`: Specifies comma-separated destination names. Valid destination names are the ones already configured.

> **Note:** Only destination names configured under the `destination` section can be used here.

- `routingRules`: List of routing rules
  - `routingRule`: Specifies the routing rule that defines the data flow. Data received on the source will be distributed to the destinations in the destination pool.
    - `name`: Specifies a unique name that identifies the routing rule.
    - `sourceName`: Specifies the source name that is configured in `sources.`
    - `destinationPoolName`: Specifies the name of the destination pool that is configured in `destinationPools.`
    - `routingPolicy`: Specifies the routing policy algorithm. Valid routing policies are `RoundRobin, WeightedRoundRobin`, and `AggregationPreferred` for `syslog` type. For `file` type, only `RoundRobin` is supported.
    - `enabled`: Specifies activation of the routing rule if set to `true`. Otherwise, the routing rule will not be applied.
    - `additionalParameters/properties`
      - `listener`: Specifies the type of `listener`, which currently includes the `syslog.address.prepend.mode` property. This property allows Load Balancer to detect IPv4 addresses, IPv6 addresses, Solaris-style addresses, and hostnames. It will add the Load Balancer's current time and the remote socket address which sent the event, if needed.

        The available values are `disabled` (the default—no information is added), `scan` (only adds information if Load Balancer does not detect an address), and `always` (always adds information).

        > **Note:** Using the `scan` value will have a negative performance impact. Only enable `scan` mode if necessary. It is better to use routing rules with `disabled` or `always` options, if it is known that the sources will always have or not have addresses included.

`statisticsLogging`

- `logInterval`: Specifies the statistics logging interval in milliseconds. By default, the statistics are logged every minute (60,000 ms).

`webServer`

> **Note:** This configuration is required per Load Balancer installation now and will be enabled in a future release

- `httpsPort`: Specify the HTTPS port. By default, it uses 8443 as the listening port.

`globalParameters`

- `batch.buffersize:` Specifies the maximum buffer size in bytes that can be used for the batch criteria. Load Balancer creates an event batch right before the total event size limit is reached.
- `batch.eventcount:` Specifies the total number of events that can be used as the batch cut-off criteria.
- `batch.timeout:` Specifies the timeout in milliseconds. A new batch will be created if the time reaches this value after the last batching and at least one event is waiting in the buffer.

  Note: Load Balancer applies these three batch parameters together using whichever condition is met first.

- `trust.store.relative.location:` Specifies the location of the trust store as a relative path in relation to the Load Balancer installation. The default value is `jre/lib/security/`, which translates to, for example: `$ARCSIGHT_HOME/jre/lib/security`
- `trust.store.name:` Specifies the name of the trust store specified under `trust.store.relative.location`. The default location is `cacerts`, which internally translates to, for example: `$ARCSIGHT_HOME/jre/lib/security/cacerts` (using the trust store relative location.)
- `trust.store.password:` Specifies the password for the trust store.
- `destination.monitoring.interval.ms:` Specifies the time interval in milliseconds for the destinations being monitored. This applies to all destinations across all destination pools. The status and health information of the destination is queried once every time interval. The default value is `60,000ms = 1 minute`. It is not recommended to reduce this time period as excessive destination querying impairs Load Balancer's performance and SmartConnectors refresh this information once a minute.
- `weighted.interval.max:` Specifies the maximum time interval before re-calculating the load distribution for Weighted Round Robin mode. This should be at least a few times the target interval.
- `weighted.interval.target:` Specifies the ideal time interval before re-calculating the load distribution for Weighted Round Robin mode. As event rates through Load Balancer always vary slightly, this will not be an exact number.
- `aggregation.connector.event.ratio:` Specifies the ratio of EPS-to-queue rate below which sources will be reallocated away from a connector in Aggregation Preferred mode. This is specified as a ratio, for example, 0.9 to indicate 90%.
- `aggregation.connector.fail.periods:` Specifies the number of consecutive monitoring intervals in which a connector must fail the above check before action is taken in

Aggregation Preferred mode to prevent transient issues from causing unneeded reallocations. Note that this is based on the statistics logger configuration, not `destination.monitoring.interval.ms`.

- `aggregation.reallocated.warn.ratio`: Specifies the ratio of the number of connectors that must be reallocated in Aggregation Preferred mode before a warning message is sent to the log file that says there is insufficient capacity. This is specified as a ratio, for example, 0.9 to indicate 90%.
- `aggregation.reallocation.max.ratio`: Specifies the maximum ratio of events from the previous monitoring interval to reallocate away from a connector in a single monitoring interval in Aggregation Preferred mode. This is specified as a ratio, for example, 0.9 to indicate 90%. Reallocation of sources continues until either this or the `aggregation.reallocation.max.sources` limit is reached, depending on which occurs first.
- `aggregation.reallocation.max.sources`: Specifies the maximum number of sources to reallocate away from a connector in a single monitoring interval in Aggregation Preferred mode. Reallocation of sources continues until either this or the `aggregation.reallocation.max.ratio` limit is reached, whichever occurs first.
- `queue.max.consumer`: (TCP only) Specifies the maximum queue size for the per event source buffer, before batches are created, as a number of events. If there are a number of sources each sending a small amount of events, this should be somewhat low. If there are a few sources each sending a large amount of events, this should be somewhat high. Tune this to be roughly 1-2 seconds' worth of events, given the batching parameters and expected event rate. Note that these queues are usually nearly empty and are just to absorb load. Events that are in this queue are not persisted anywhere and will be lost if Load Balancer terminates unexpectedly. When this queue is full, no further events are read off of the socket, so event sources will experience TCP backpressure.
- `queue.max.producer`: Specifies the maximum queue size for the per destination asynchronous buffer as a number of batches (TCP) or events (UDP). While this buffer is full, TCP will buffer batches on disk up to `persistent.queue.disk.limit` bytes, and UDP will buffer events in memory up to the `udp.events.queue.capacity` value before starting to drop events. Events in this queue are still persisted on disk for TCP.
- `udp.consumer.threads`: Specifies the number of threads used to read UDP packets from the network.
- `tcp.consumer.threads`: Specifies the number of threads to use to dispatch TCP event batches. If files are appearing in the `$ARCSIGHT_HOME/user/loadbalancer/lbdata/persistence/RULE-NAME` directory faster than they are disappearing, consider increasing this value, up to the number of destinations in a single destination pool.
- `persistent.queue.disk.limit`: The maximum size, in bytes, of the persisted event batches for each TCP routing rule. This includes internal overhead, but not file system overhead. For

example, if you have two TCP routing rules, you need at least two times this much disk space available, plus a margin for file system metadata and to account for block sizes. This should be at least 1073741824 (1 GiB), and must be at least twice the expected throughput per second.

- `persist`: Persist event batch on disk for TCP mode. Defaults to `true`. Disable to remove the disk as a bottleneck, but all events in-flight in Load Balancer will be lost if it is shut down.
- `load.expression.default`: Specifies custom load-level calculation expressions as a global default override for all destinations (excluding those which do not have their own per-destination expression.) For more information, see "Calculating Loads for Routing" on page 89.
- `ssl.enabled.protocols`: Enabled TLS protocols. Defaults are `TLSv1,TLSv1.1,TLSv1.2`.
- `ssl.cipher.suites`: Enabled cipher suites for TLS connections. Defaults are `TLS_RSA_WITH_AES_128_CBC_SHA,SSL_RSA_WITH_3DES_EDE_CBC_SHA`.
- `ssl.keystore.password`: Password for the `ssl.keystore.file`. Note: This currently must be in plain text. The default is `changeit`.
- `ssl.key.password`: Password for the key in the `ssl.keystore.file`. Note: This currently must be in plain text. The default is `changeit`.
- `ssl.keystore.file`: Keystore file for remote management and TLS syslog listeners. If this file does not exist, it will be automatically created with the parameters described below, and also overwrite the `ssl.cert.file` with the newly generated certificate. This path is relative to `ARCSIGHT_HOME/user/loadbalancer`. The default is `loadbalancer.p12`.
- `ssl.cert.file`: Certificate file for remote management and TLS syslog listeners. If this file does not exist, it will be automatically created by exporting the certificate from `ssl.keystore.file`. This path is relative to `ARCSIGHT_HOME/user/loadbalancer`. The default is `loadbalancer.cer`.
- `ssl.cert.validity`: How long, in days, an automatically generated certificate should be valid. The default is `3650`.
- `ssl.cert.key.size`: SSL key size, in bits. The default is: `2048`.
- `ssl.cert.organization`: "O" (organization) field in the distinguished name ("DN") of an automatically generated certificate. The default is `ArcSight`.
- `ssl.cert.organizational.unit`: "OU" (organizational unit) field in the DN of an automatically generated certificate. The default is `loadbalancer`.
- `ssl.cert.locality`: "L" (locality/city) field in the DN of an automatically generated certificate. The default is `NA`.
- `ssl.cert.state`: "ST" (state) field in the DN of an automatically generated certificate. The default is `NA`.
- `ssl.cert.country`: "C" (state) field in the DN of an automatically generated certificate. The default is `US`.
- `reload.configuration`: Specifies whether the configuration can be reloaded from primary node in primary-secondary mode. You must set this value in the configuration file of the primary node. Do

not modify this value in the secondary node.

To change the configuration in primary-secondary mode without downtime:

a. Shut down the primary node.

b. Set the value of `reload.configuration` to `true`.

c. Modify the value of configuration.

Note: You can not modify the value of `memberhosts`.

d. Start the primary node.

`clusterconfigurations`

`hazelcast.max.no.heartbeat.seconds`: Specifies the timeout interval in seconds, for a node to assume that it is not reachable. The default value is: `300 seconds`. If any event loss is observed during the fail-over, you can reduce this timeout interval for a faster fail-over.

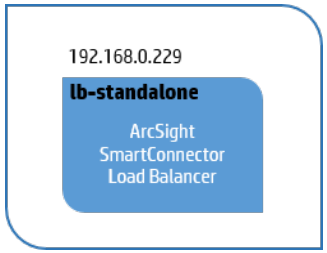
Example:

```
<hazelCastParameters>
<properties>
<property key="hazelcast.max.no.heartbeat.seconds" value="60"/>
</properties>
</hazelCastParameters>
```

Configuration Examples

Configuring MemberHosts in Standalone Mode

Load Balancer can be configured to run in standalone mode as shown in the diagram.

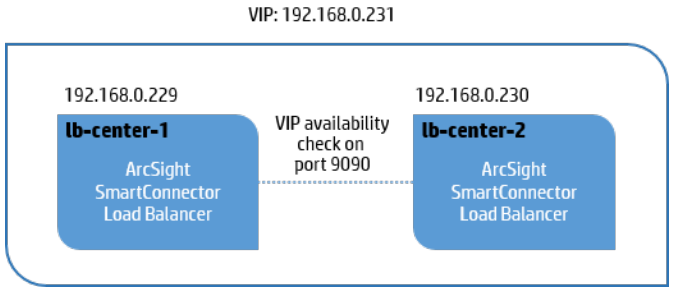



```
   <memberHosts vipAddress="192.168.0.229" vipPingPort="9090">
      <memberHost name="lb-standalone" host="192.168.0.229" port="6702" isPrimary="false"
vipBindCommand="sudo /sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="sudo
/sbin/ifdown /etc/sysconfig/network-scripts/ifcfg-eth0:1"/>
      </memberHosts>
  <memberIdentity>lb-standalone</memberIdentity>
```

Note: `vipBindCommand` and `vipUnbidCommand` have `sudo` in the command because Load Balancer is not running as root.

Configuring MemberHosts as Peer

When Load Balancer is deployed to run as peer, Load Balancer is installed on two hosts sharing the same virtual IP address. In the diagram below
lb-center-1 and lb-center-2 are running as peer. The member host that starts first will run as the active member and pushes the configuration value to the other member host. Note that `isPrimary` is set to false in both configurations.



For lbConfig.xml on **lb-center-1**:

```
  <memberHosts vipAddress="192.168.0.231" vipPingPort="9090">
    <memberHost name="lb-center-1" host="192.168.0.229" port="6702" isPrimary="false"
vipBindCommand="sudo /sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="sudo
/sbin/ifdown /etc/sysconfig/network-scripts/ifcfg-eth0:1"/>
   </memberHosts>
   <memberHost name="lb-center-2" host="192.168.0.230" port="6702" isPrimary="false" \
     vipBindCommand="sudo /sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" \
     vipUnbindCommand="sudo /sbin/ifdown /etc/sysconfig/network-scripts/ifcfg-eth0:1"/>
   </memberHosts>
```

```
   <memberIdentity>lb-center-1</memberIdentity>
```

For lbConfig.xml on **lb-center-2**:

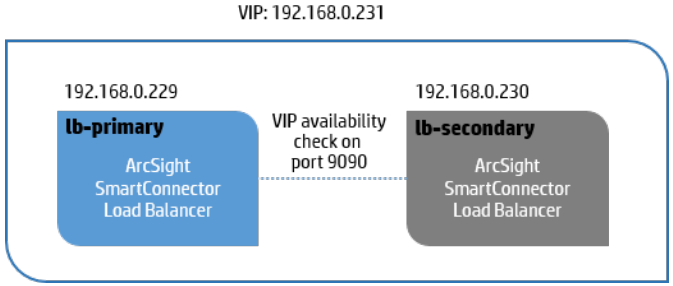
```
   <memberHosts vipAddress="192.168.0.231" vipPingPort="9090">
       <memberHost name="lb-center-1" host="192.168.0.229" port="6702" isPrimary="false"
vipBindCommand="sudo /sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="sudo
/sbin/ifdown /etc/sysconfig/network-scripts/ifcfg-eth0:1"/>
       </memberHosts>
       <memberHost name="lb-center-2" host="192.168.0.230" port="6702" isPrimary="false"
vipBindCommand="sudo /sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="sudo
/sbin/ifdown /etc/sysconfig/network-scripts/ifcfg-eth0:1"/>
       </memberHosts>
   <memberIdentity>lb-center-2</memberIdentity>
```

Note: `vipBindCommand` and `vipUnbidCommand` have `sudo` in the command because Load Balancer is not running as root.

Configuring MemberHosts as Primary-Secondary

When Load Balancer is going to be deployed as primary-secondary, Load Balancer is installed in two hosts sharing the same virtual IP address. In the diagram below lb-primary is designated as the primary load balancer and the value for `isPrimary` is set to `true` for `memberHost`, while it is set to `false` for lb-secondary. Always start lb-primary member host first to have configuration synchronized to lb-secondary member host.

Note: `vipBindCommand` and `vipUnbidCommand` do not have `sudo` in the command because Load Balancer is running as root.



For lbConfig.xml on **lb-primary**:

```
    <memberHosts vipAddress="192.168.0.231" vipPingPort="9090">
        <memberHost name="lb-primary" host="192.168.0.229" port="6702" isPrimary="true"
vipBindCommand="/sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="/sbin/ifdown
/etc/sysconfig/network-scripts/ifcfg-eth0:1"/>
        </memberHosts>
        <memberHost name="lb-center-2" host="192.168.0.230" port="6702" isPrimary="false"
vipBindCommand="/sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="/sbin/ifdown
/etc/sysconfig/network-scripts/ifcfg-eth0:1"/>
        </memberHosts>
    <memberIdentity>lb-primary</memberIdentity>
```

For lbConfig.xml on **lb-secondary**:

```
    <memberHosts vipAddress="192.168.0.231" vipPingPort="9090">
        <memberHost name="lb-center-1" host="192.168.0.229" port="6702" isPrimary="true"
vipBindCommand="/sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="/sbin/ifdown
/etc/sysconfig/network-scripts/ifcfg-eth0:1"/>
        </memberHosts>
```

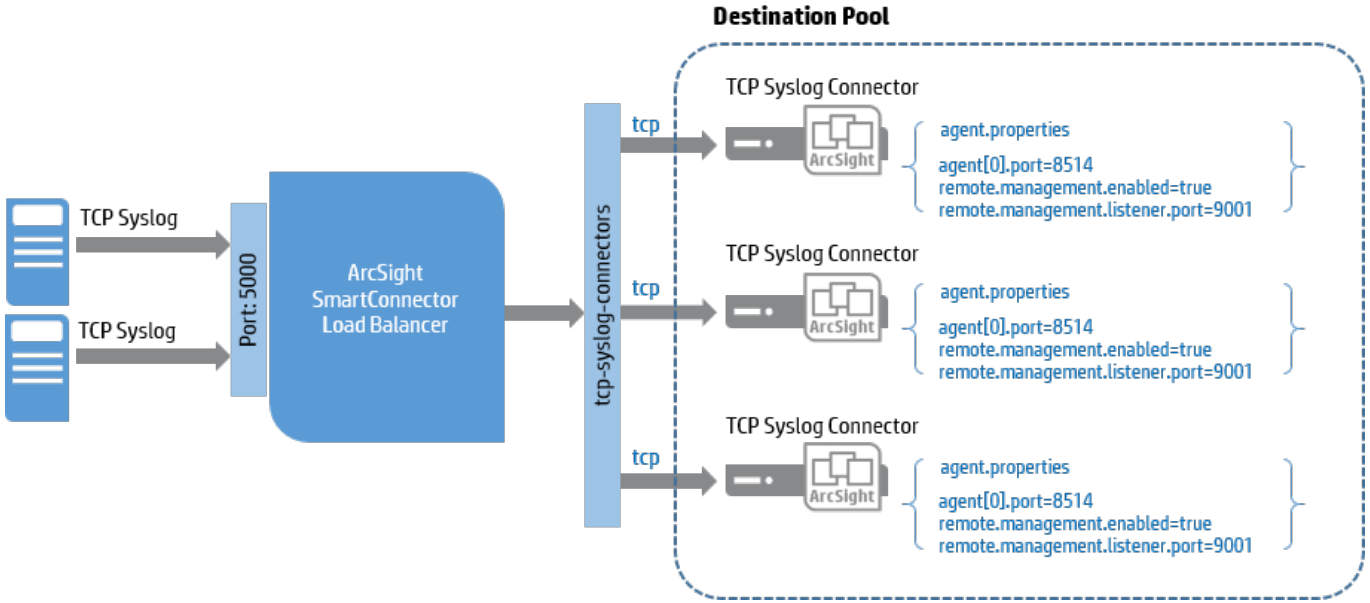
```
        <memberHost name="lb-secondary" host="192.168.0.230" port="6702" isPrimary="false"
vipBindCommand="/sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="/sbin/ifdown
/etc/sysconfig/network-scripts/ifcfg-eth0:1"/>

        </memberHosts>

   <memberIdentity>lb-secondary</memberIdentity>
```

Syslog Load Balancing Routing Rule Example

The following diagram illustrates two syslog servers feeding an input stream into Load Balancer on port 5000 using a TCP connection. In this scenario, Load Balancer distributes the events to three TCP syslog connectors which are grouped as one destination pool called 'tcp-syslog-connectors'. The following configuration file shows an example of how to configure the routing rule used in this scenario. Note that `remote.management.listener.port` is configured per destination. This information is used to detect the health and load of the connectors in destination pool and the connector is considered down if it is configured with an incorrect value.




```
<routing>
   <destinationPools>
      <destinationPool name="tcp-syslog-connectors"
         destinations="tcp-syslog-1,tcp-syslog-2,tcp-syslog-3"/>
   </destinationPools>
   <destinations>
      <destination name="tcp-syslog-1" type="syslog" host="192.168.0.1"
            port="8514" protocol="tcp">
         <additionalParameters type="connector">
```

```
      <properties>
         <property key="remote.management.listener.port" value="9001"/>
      </properties>
   </additionalParameters>
</destination>
<destination name="tcp-syslog-2" type="syslog" host="192.168.0.2"
      port="8514" protocol="tcp">
   <additionalParameters type="connector">
      <properties>
         <property key="remote.management.listener.port" value="9001"/>
      </properties>
   </additionalParameters>
</destination>
<destination name="tcp-syslog-3" type="syslog" host="192.168.0.3"
      port="8514" protocol="tcp">
   <additionalParameters type="connector">
      <properties>
         <property key="remote.management.listener.port" value="9001"/>
      </properties>
```

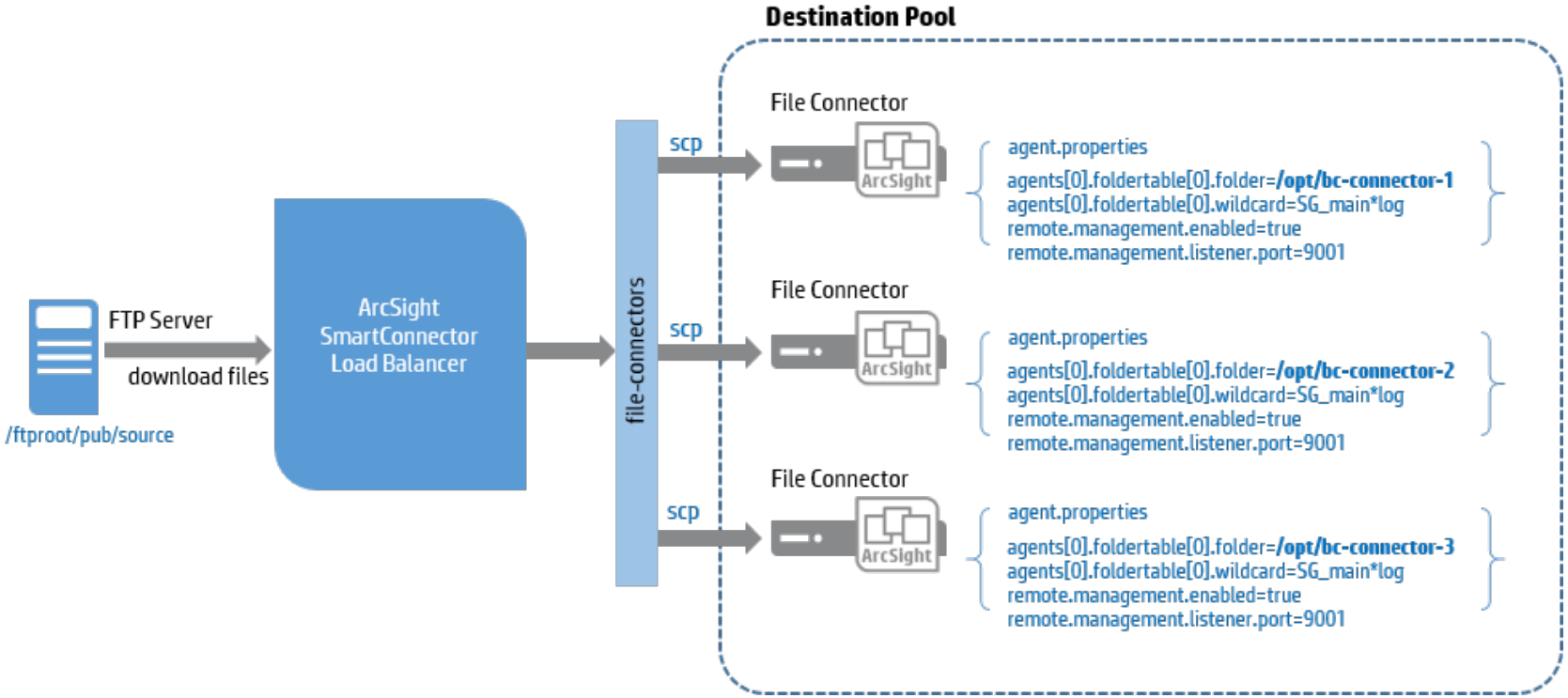
```
            </additionalParameters>
         </destination>
      </destinations>
      <routingRules>
         <routingRule name="firewall-syslog" sourceName="syslog-receiver"
            destinationPoolName="tcp-syslog-connectors"
            routingPolicy="WeightedRoundRobin" enabled="true">
               <additionalParameters type="listener">
                 <properties>
                   <property key="syslog.address.prepend.mode" value="scan"/>
                 </properties>
               </additionalParameters>
        </routingRule>
      </routingRules>
      <sources>
         <source name="syslog-receiver" type="syslog" port="5000"
            protocol="tcp"/>
      </sources>
   </routing>
```

File Load Balancing Routing Rule Example

The following diagram shows a configuration example of a routing rule for file load balancing. To define the source, an FTP host address, the credentials, and the path need to be defined. Here, the FTP root directory is `/ftproot/pub`, so the actual location to FTP client should be the `source` directory. The rest of the routing rule configuration is similar to syslog routing rule configuration.

Note: When uploading files to the source FTP server, be sure to use a temporary file name and specify the filter in the `fileFilter` parameter to filter out temporary files. After file upload is complete, rename the file to an original name.

Note: The agent name for file connectors is specified in this configuration.




```
<routing>
   <destinationPools>
      <destinationPool name="file-connectors"
         destinations="file-connector-1,file-connector-2,file-connector-3"/>
   </destinationPools>
   <destinations>
      <destination name="file-connector-1" type="file"
```

```
   path="/opt/bc-connector-1" host="192.168.0.1" protocol="scp"
   username="admin" password="password"
   knownHostsFile="/home/arcsight/.ssh/known_hosts">
   <additionalParameters type="connector">
      <properties>
         <property key="remote.management.listener.port" value="9001"/>
         <property key="agent.name" value="bc-connector-1"/>
      </properties>
   </additionalParameters>
</destination>
<destination name="file-connector-2" type="file"
   path="/opt/bc-connector-2" host="192.168.0.2" protocol="scp"
   username="admin" password="password"
   knownHostsFile="/home/arcsight/.ssh/known_hosts">
   <additionalParameters type="connector">
      <properties>
         <property key="remote.management.listener.port" value="9001"/>
         <property key="agent.name" value="bc-connector-2"/>
      </properties>
```

```
      </additionalParameters>
   </destination>
   <destination name="file-connector-3" type="file"
      path="/opt/bc-connector-3" host="192.168.0.3" protocol="scp"
      username="admin" password="password"
      knownHostsFile="/home/arcsight/.ssh/known_hosts">
      <additionalParameters type="connector">
         <properties>
            <property key="remote.management.listener.port" value="9001"/>
            <property key="agent.name" value="bc-connector-3"/>
         </properties>
      </additionalParameters>
   </destination>
</destinations>
<routingRules>
   <routingRule name="file-rule" sourceName="file-watcher"
      destinationPoolName="file-connectors" routingPolicy="RoundRobin"
      enabled="true"/>
</routingRules>
```

```
   <sources>
      <source name="file-watcher" type="file" path="/source"
         host="192.168.1.225" protocol="ftp" username="admin"
         password="OBFUSCATE.1:B8R3Ts5XXui0aBjFn1Js7Q=="
         moveToDirectory=".done" fileFilter="SG_main.*log"
         localWorkDirectory="/tmp" recursive="true" passive="false" />
      </sources>
   </routing>
```

Sample Configuration File

The following example shows the Load Balancer configuration file configured for syslog load balancing. The template provided by Load Balancer is shown in Appendix A.

Note: This is a sample configuration file with passwords obfuscated since it was captured after Load Balancer started.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>


<lbConfiguration>
       <memberHosts vipAddress="192.168.5.231" vipPingPort="9090">
               <memberHost name="primary-node" host="192.168.5.229" port="7701" isPrimary="false" vipBindCommand="sudo /sbin/ifup
/etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="sudo /sbin/ifdown /etc/sysconfig/network-scripts/ifcfg-
eth0:1"/>
               <memberHost name="secondary-node" host="192.168.5.230" port="7701" isPrimary="false" vipBindCommand="/sbin/ifup
/etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="/sbin/ifdown /etc/sysconfig/network-scripts/ifcfg-
eth0:1"/>
       </memberHosts>
```

```
<memberIdentity>primary-node</memberIdentity>
<notification enable="true">
        <enabledNotification>
                <event name="MemberHostUp" message="Member node is up"/>
                <event name="MemberHostDown" message="Member node is down"/>
                <event name="DestinationUp" message="Destination is up"/>
                <event name="DestinationDown" message="Destination is down"/>
        </enabledNotification>
        <email>
                <prefix>[Load Balancer]</prefix>
                <recipients>nanjoo.ban@hp.com</recipients>
                <sender>nanjoo.ban@hp.com</sender>
                <smtpServer>englab-mail.arst.usa.hp.com</smtpServer>
        </email>
</notification>
<routing>
        <destinationPools>
                <destinationPool name="tcp-syslog-connectors" destinations="tcp-syslog-1,tcp-syslog-2"/>
                <destinationPool name="bc-file-connectors" destinations="bc-connector-1,bc-connector-2"/>
        </destinationPools>
        <destinations>
                <destination name="tcp-syslog-1" type="syslog" host="192.168.0.11" port="8514" protocol="tcp">
                        <additionalParameters type="connector">
                                <properties>
                                        <property key="remote.management.listener.port" value="9001"/>
                                </properties>
                        </additionalParameters>
                </destination>
                <destination name="tcp-syslog-2" type="syslog" host="192.168.0.12" port="8514" protocol="tcp">
                        <additionalParameters type="connector">
                                <properties>
                                        <property key="remote.management.listener.port" value="9001"/>
                                </properties>
                        </additionalParameters>
                </destination>
                <destination name="bc-connector-1" type="file" path="/opt/ArcSightSmartConnectors/bc-connector/
```

```
                        current/file-feeds" host="192.168.0.1" protocol="scp" username="admin"
password="OBFUSCATE.1:wAeVeZlW8m4Cq2wLEupkjA==" recursive="false" flatten="false" passive="false"
knownHostsFile="/home/arcsight/.ssh/known_hosts">
                            <additionalParameters type="connector">
                                <properties>
                                    <property key="remote.management.listener.port" value="9001"/>
                                    <property key="agent.name" value="bc-connector"/>
                                </properties>
                            </additionalParameters>
                    </destination>
                    <destination name="bc-connector-2" type="file" path="/opt/ArcSightSmartConnectors/bc-connector/

                        current/file-feeds" host="192.168.0.2" protocol="scp" username="admin"
password="OBFUSCATE.1:wAeVeZlW8m4Cq2wLEupkjA==" recursive="false" flatten="false" passive="false"
knownHostsFile="/home/arcsight/.ssh/known_hosts">
                            <additionalParameters type="connector">
                                <properties>
                                    <property key="remote.management.listener.port" value="9001"/>
                                    <property key="agent.name" value="bc-connector"/>
                                </properties>
                            </additionalParameters>
                    </destination>
            </destinations>
            <routingRules>
                    <routingRule name="syslog-tcp-rule" sourceName="syslog-tcp" destinationPoolName="tcp-syslog-connectors"
routingPolicy="WeightedRoundRobin" enabled="true">
                            <additionalParameters type="listener">
                                <properties>
                                    <property key="syslog.address.prepend.mode" value="scan"/>
                                </properties>
                            </additionalParameters>
                     </routingRule>

                     <routingRule name="bc-file-rule" sourceName="bc-ftp-server" destinationPoolName="bc-file-connectors"
routingPolicy="RoundRobin" enabled="true"/>

             </routingRules>
            <sources>
```

```
                        <source name="syslog-tcp" type="syslog" host="192.168.8.12" port="8002" protocol="tcp"/>
                        <source name="bc-ftp-server" type="file" path="bc-files" host="192.168.8.34" protocol="ftp" username="arcsight"
password="OBFUSCATE.1:y05cvjSnFlVybZFBBEOHiQ==" recursive="true" flatten="true" passive="true"
localWorkDirectory="/tmp" fileFilter=".*log"/>
                </sources>
        </routing>
        <statisticsLogging logInterval="60000" />
        <webServer httpsPort="8443" certificatePath="loadbalancer.cer" keystorePath="loadbalancer.p12"/>
</lbConfiguration>
```

Starting Load Balancer

Start Load Balancer from the host that has been fully configured first (primary or active) to allow the configuration to correctly sync to the passive or secondary node. Otherwise, the configuration information will not be passed properly if the passive or secondary host is started first.

To start Load Balancer:

1. Go to the `$ARCSIGHT_HOME` directory and start Load Balancer with the following command:

   ```
   # bin/arcsight loadbalancer
   ```

2. Log on to the secondary machine, if one exists, and start Load Balancer.

   Note: In primary-secondary HA mode, if the secondary node starts before the primary node, the configuration will not be copied over properly.

Installing Load Balancer as a Service

To run Load Balancer as a service, the initial installation must be completed and there must be a working configuration file. Validate the working configuration file by running Load Balancer as a standalone application, as shown in "Starting Load Balancer" above.

To install the files needed to run Load Balancer as a service:

1. Go to the `$ARCSIGHT_HOME` directory and run the following command as root:

   ```
   # bin/arcsight loadbalancer_service –i
   ```

   This command installs the files necessary for running Load Balancer as a service.

2. Run the command without any switches to see the usage:

   # bin/arcsight loadbalancer_service

   ServiceTool - ArcSight SmartAgent Service Tool

   Version : 1.0

   Confidential commercial computer software. Valid license required.

   Usage: ServiceTool <parameters>

   Optional Parameters:

   -sd <description> Service/Script description (Install only) (Load Balancer for Arcsight SmartConnectors)

   -sn <name> Service/Script name (Install only) (connlb)

   Options:

-h help - Get help for this command

-i install - Installs the SmartConnectors LoadBalancer as a service

-r remove - Removes the SmartConnectors LoadBalancer Service

Note: If you are not the root user, an error message displays when invoking service tool:

```
# bin/arcsight loadbalancer_service
 Assuming ARCSIGHT_HOME: /home/arcsight/beta1/current
 Assuming JAVA_HOME: /home/arcsight-1/beta1/current/jre

 ArcSight Load Balancer Service Tool starting...
 ************************************************
 ERROR:This program should be run as [root]. Exiting...
 ************************************************
```

3. After the service `arc_connlb` is created, it can be accessed with service commands. For example:

```
# service arc_connlb status

Running as root

Output will be logged to $ARCSIGHT_HOME/current/logs/lb.out.wrapper.log

Getting status of ArcSight Load Balancer for Arcsight SmartConnectors...

ArcSight Load Balancer for Arcsight SmartConnectors is not running.
```

4. (Optional) To give the service a different name, use the `–sn` switch during service installation. The line below shows the service name changed to `arc_loadbalancer`. (The `arc_` is added before all services names.) If no other name is suggested, the default service name is `'connlb'`.

```
# bin/arcsight loadbalancer_service -sn loadbalancer -i
```

5. Use the `–sd` switch to give a different service description. For example, change the service description to 'LBService':

```
# bin/arcsight loadbalancer_service -i -sd LBService
```

6. Using the service description change shown step 5, the status command displays as follows:

```
# service arc_connlb status

Running as root

Output will be logged to $ARCSIGHT_HOME/current/logs/lb.out.wrapper.log
```

```
Getting status of ArcSight LBService...

ArcSight LBService is not running.
```

7. To remove the service files, use the following command:

```
# bin/arcsight loadbalancer_service -r
```

Starting or Stopping the Load Balancer Service

After you have installed Load Balancer as a service, you can start or stop the service at any time. You are expected to be a `root` user to run Load Balancer as a service.

To start or stop the Load Balancer service:

> **Note:** Only root users may run Load Balancer as a service.

1. To start the Load Balancer service, use the following command:

```
# /etc/init.d/arc_connlb start
```

or

```
service arc_connlb start
```

> **Note:** If you changed the default service name (connlb), use that name in place of 'connlb'.

2. To stop the Load Balancer service, use the following command:

```
# /etc/init.d/arc_connlb stop
```

or

```
service arc_connlb stop
```

Load Balancer Service Commands

Other commands available when running Load Balancer as a service include:

- `start` — Starts Load Balancer as a service.
- `stop` — Stops Load Balancer as a service.
- `restart` — Stops the service, if it's running, then restarts it.
- `dump` — Captures the current JVM state including all the running threads and their states. The output will be present in `lb.out.wrapper.log`. The Load Balancer service continues to run normally after the dump. This command needs Load Balancer to be running or the command will have no effect.

- `console` — Runs the Load Balancer service as an application from the current window, which can be stopped with a Ctrl + c or with the `stop service` command from another window. The log will be displayed on the console.

Usage is displayed if no command is given. For example:

service arc_connlb

Running as root

Output will be logged to $ARCSIGHT_HOME/current/logs/lb.out.wrapper.log

Usage: $ARCSIGHT_HOME/current/bin/lb.wrapper.sh { start | stop | restart | dump | status | console }

Load Balancer Service-related Logs

The log will be redirected to `lb.out.wrapper.log` under the `logs` directory.

Interpreting Logs

Statistics are divided by:

- Routing rule—each rule is on its own line
- Locations—from sources or to destinations
- Metrics—bytes, events, and batches
- Time units—average per second SLC, total SLC, and total since startup

For each routing rule, the combined totals from every source are listed first, followed by the combined totals to every destination, and the individual statistics per destination.

Overall statistics:

2015-08-24 08:43:25,556 [INFO][statisticsLogging][com.arcsight.lb.stats.StatLoggingTask][run] - Load Balancer statistics {metric=average per second-SLC/SLC/Total}:

2015-08-24 08:43:25,557 [INFO][statisticsLogging][com.arcsight.lb.stats.StatLoggingTask][run] - Routing rule=[syslog-tcp-rule-1]: [src= (total),bytesRcvd=1636923/16369232/188733127,eventsRcvd=10000/100000/1153000,batchesRcvd =10/100/1153], [dest= (total),bytesSent=1636923/16369232/196591275,eventsSent=10000/100000/1201000,batchesSent= 10/100/1201],[dest=tcp-syslog- 1,bytesSent=818452/8184521/98377554,eventsSent=5000/50000/601000,batchesSent=5/50/601], [dest=tcp-syslog- 2,bytesSent=818471/8184711/98213721,eventsSent=5000/50000/600000,batchesSent=5/50/600]]

2015-08-24 08:43:25,557 [INFO][statisticsLogging][com.arcsight.lb.stats.StatLoggingTask][run] - Routing rule=[syslog-tcp-rule-1]: In: EPS=[10000] Bytes/s=[1636923] Out: EPS=[10000] Bytes/s=[1636923]

Chapter 3 — Load Balancer REST API

Load Balancer provides an Application Programming Interface (API) for programmatic access to Load Balancer resources. You can use Standard APIs to configure Load Balancer.

Configuration

This section describes how to configure Load Balancer using Rest API. The following configurations must be done in the same sequence:

1. Source

2. Destination

3. Destination pools

4. Routing rules

When Load Balancer is configured to run in HA mode, you can execute REST APIs either using the virtual IP or the actual IP address of the primary/active node. Requests to the secondary/standby node will not be successful.

The following element is unique to each node. You must add it to both primary and secondary nodes.

```
<webServer httpsPort="8443" certificatePath="config/loadbalancer.cer"
keystorePath="config/loadbalancer.p12"/>
```

The following values used in this element are relative to ARCSIGHT_HOME:

- certificatePath: Location of the certificate file.
- keystorePath: Location of the keystore file.

Load Balancer API Reference

Retrieving a List of Routing Rules

This method displays all the routing rules.

API Reference

```
GET /config/routingRules
```

Sample Request

URI: `GET https://127.0.0.1:8443/config/routingRules`

Sample Response

Success Code: 200 (OK)

Body:


```
[
{
"name": "syslog-tcp-rule",
"sourceName": "syslog-tcp",
"destinationPoolName": "tcp-syslog-connectors",
"routingPolicy": "RoundRobin",
"enabled": true,
"additionalParameters": {
"type": "listener",
"properties": {
"property": [
{
"key": "syslog.address.prepend.mode",
"value": "scan"
}
]
}
}
},
{
"name": "syslog-udp-rule",
"sourceName": "syslog-udp",
"destinationPoolName": "udp-syslog-connectors",
"routingPolicy": "WeightedRoundRobin",
```

```
"enabled": true
}
]
```

Retrieving Details of a Routing Rule

This method displays details of the selected routing rule.

API Reference

```
GET /config/routingrules/routingrule/<name of the routing rule>
```

Sample Request

URI: `GET https://127.0.0.1:8443/config/routingrules/routingrule/syslog-tcp-rule`

Sample Response

Status: 200 (OK)

Body:

```
{
"name": "syslog-tcp-rule",
"sourceName": "syslog-tcp",
"destinationPoolName": "tcp-syslog-connectors",
"routingPolicy": "RoundRobin",
"enabled": true,
"additionalParameters": {
"type": "listener",
"properties": {
"property": [
{
"key": "syslog.address.prepend.mode",
"value": "scan"
}
]
}
```

```
}
}
```

Error Code

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Routing rule not found, routing rule=[syslog-tcp-rule-non-existent]"
}
]
```

Reason: This error occurs when you try to retrieve a routing rule that is not present.

Creating a Routing Rule

This method adds a new routing rule.

API Reference

```
POST /config/routingrules/routingrule
```

Content-Type

application/json

Sample Request

URI: `POST https://127.0.0.1:8443/config/routingrules/routingrule`

Body:

```
{
"name": "syslog-udp-rule",
"sourceName": "syslog-udp",
"destinationPoolName": "udp-syslog-connectors",
"routingPolicy": "WeightedRoundRobin",
"enabled": false
```

```
}
```

Sample Response

Status: 201 (Created)

Body:

```
{
"name": "syslog-udp-rule",
"sourceName": "syslog-udp",
"destinationPoolName": "udp-syslog-connectors",
"routingPolicy": "WeightedRoundRobin",
"enabled": false
}
```

Error Codes

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Routing rule not found, routing rule=[syslog-tcp-rule-non-existent]"
}
]
```

Reason: This error occurs when the routing rule in the request contains a source name that is already referred by some other routing rule.

Status: 400 (Bad Request)

Body:

```
[
{
"description": "Duplicate name found for Routing Rule=[syslog-tcp-rule]",
"errorSource": "Configuration"
```

```
}
]
```

Reason: This error occurs when the routing rule in the request contains a name that is already present in some other routing rule.

Status: 400 (Bad Request)

Body:

```
[
{
"description": "Destination pool undefined: routing rule=[syslog-udp-rule], destination pool=[udp-syslog-connectors]",
"errorSource": "Configuration"
}
]
```

Reason: This error occurs when the destination pool mentioned in the request is not present in `lbConfig.xml`.

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration"
"description": "UDP Sources or destinations may only be used with other UDP sources and destinations: Routing Rule=[syslog-tcp-rule2]"
}
]
```

Reason: This error occurs when the protocol of the source is not similar to the protocol of the destination used in the destination pool of the request.

Status: 400 (Bad Request)

Body:

```
[
{Can not construct instance of com.arcsight.lb.bean.RoutingPolicy from String value
```

```
'WeightedRoundRobin-NoExist': value not one of declared Enum instance names:

[RoundRobin, AggregationPreferred, WeightedRoundRobin] at [Source:
org.glassfish.jersey.message.internal.ReaderInterceptorExecutor$UnCloseableIn
putStream

@66ee8a07; line: 7, column: 47] (through reference chain:
com.arcsight.lb.bean.RoutingRule["routingPolicy"])
```

Reason: This error occurs when the routing policy mentioned in the request is not one these: RoundRobin, AggregationPreferred, and WeightedRoundRobin.

Deleting a Routing Rule

This method deletes the selected routing rule.

API Reference

```
DELETE /config/routingrules/routingrule/<name of the routing rule to be
deleted>
```

Sample Request

URI: `DELETE https://127.0.0.1:8443/config/routingrules/routingrule/syslog-udp-rule1`

Sample Response

Status: 200 (OK)

Body:

```
{

"name": "syslog-udp-rule",

"sourceName": "syslog-udp",

"destinationPoolName": "udp-syslog-connectors",

"routingPolicy": "WeightedRoundRobin",

"enabled": false

}
```

Error Codes

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Routing rule not found, routing rule=[syslog-udp-rule1]"
}
]
```

Reason: This error occurs when you try to delete a routing rule that is not present.

Status: 400 (Bad Request)

Body:

```
[ { "errorSource": "Configuration", "description": "Routing rule cannot be deleted while it is enabled, routing rule=[syslog-tcp-rule]" } ]
```

Reason: This error occurs when you try to delete a routing rule that is in the enabled state.

Enabling a Routing Rule

This method enables the selected routing rule.

API Reference

```
PUT /config/routingrules/routingrule/<name of the rule to be enabled>/enable
```

Sample Request

URI: `PUT https://127.0.0.1:8443/config/routingrules/routingrule/syslog-udp-rule/enable`

Sample Response

Status: 200 (OK)

Body:

```
{ "name": "syslog-udp-rule", "sourceName": "syslog-udp",
"destinationPoolName": "udp-syslog-connectors", "routingPolicy":
"WeightedRoundRobin", "enabled": false }
```

Error Code

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Routing rule is already Enabled: routing rule=[syslog-udp-rule]"
}
]
```

Reason: This error occurs when you try to enable a routing rule that is in the enabled state.

Disabling a Routing Rule

This method disable the selected routing rule.

API Reference

```
PUT /config/routingrules/routingrule/<name of the rule to be disabled>/disable
```

Sample Request

URI: `PUT https://127.0.0.1:8443/config/routingrules/routingrule/syslog-udp-rule/disable`

Sample Response

Status: 200 (OK)

Body:

```
{ "name": "syslog-udp-rule", "sourceName": "syslog-udp", "destinationPoolName": "udp-syslog-connectors", "routingPolicy": "WeightedRoundRobin", "enabled": true }
```

Error Code

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
```

```
"description": "Routing rule is already disabled: routing rule=[syslog-udp-rule]"
}
]
```

Reason: This error occurs when you try to disable a routing rule that is already in the disabled state.

Retrieving a List of Sources

This method displays all the sources.

API Reference

```
GET /config/sources
```

Sample Request

URI: `GET https://127.0.0.1:8443/config/sources`

Sample Response

Success Code: 200 (OK)

Body:

```
[
{
"name": "syslog-tcp",
"type": "SYSLOG",
"protocol": "tcp",
"port": 512
},
{
"name": "syslog-udp",
"type": "SYSLOG",
"protocol": "udp",
"port": 513
}
]
```

Retrieving Details of a Source

This method displays details of the selected source.

API Reference

```
GET /config/sources/source/<name of the source>
```

Sample Request

URI: `GET https://127.0.0.1:8443/config/sources/source/syslog-tcp`

Sample Response

Success Code: 200 (OK)

Body:

```
{
"name": "syslog-tcp",
"type": "SYSLOG",
"protocol": "tcp",
"port": 512
}
```

Creating a Source

This method adds a new source.

API Reference

```
POST /config/sources/source
```

Content-Type

application/json

Sample Request

URI: `POST https://127.0.0.1:8443/config/sources/source`

Body:

```
{
```

```
"name": "syslog-tcp",
"type": "SYSLOG",
"protocol": "tcp",
"port": 512
}
```


Sample Response

Status: 201 (Created)

Bosy:


```
{
"name": "syslog-tcp",
"type": "SYSLOG",
"protocol": "tcp",
"port": 512
}
```


Error Codes

Status: 400 (Bad Request)

Body:


```
[
{
"errorSource": "Configuration",
"description": "Duplicate name found for Source=[syslog-tcp]"
},
{
"errorSource": "Configuration",
"description": "Duplicate port found from Source=[syslog-tcp]: Port=[512]"
}
]
```


Reason: This error occurs when the source already exists and some other source has already used the port provided in the request.

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Duplicate port found from Source=[syslog-tcp2]: Port=[512]"
}
]
```

Reason: This error occurs when the port mentioned in a new source is already associated with an existing source.

Status: 400 (Bad Request)

Body:

```
Can not construct instance of com.arcsight.lb.bean.EndPointType from String
value 'syslog': value not one of declared Enum instance names: [FILE, SYSLOG,
URI]
at [Source:
org.glassfish.jersey.message.internal.ReaderInterceptorExecutor$UnCloseableIn
putStream@160485; line: 2, column: 22] (through reference chain:
com.arcsight.lb.bean.AdaptedEndPoint["type"])
```

Reason: This error occurs when the value of the field `type` is incorrect.

Deleting a Source

This method deletes the selected source.

API Reference

```
DELETE /config/sources/source/<name of the source>
```

Sample Request

URI: `DELETE https://127.0.0.1:8443/config/sources/source/syslog-tcp`

Sample Response

Status: 200 (OK)

Body:

```
{
"name": "syslog-tcp",
"type": "SYSLOG",
"protocol": "tcp",
"port": 512
}
```

Error Code

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "The source=[syslog-tcp] is being referenced by Routing Rule=
[[syslog-tcp-rule]]"
}
]
```

Reason: This error occurs when the source you are trying to delete is referenced by an existing routing rule.

Retrieving a List of Destinations

This method displays all the destinations.

API Reference

```
GET /config/destinations
```

Sample Request

URI: `GET https://127.0.0.1:8443/config/destinations`

Sample Response

Success Code: 200 (OK)

Body:

```
[
```

```
{
"name": "syslog-tcp-connector-1",
"type": "SYSLOG",
"host": "10.71.140.31",
"protocol": "tcp",
"additionalParameters": {
"type": "connector",
"properties": {
"property": [
{
"key": "remote.management.listener.port",
"value": "9001"
}
]
}
},
"port": 514
},
{
"name": "syslog-udp-connector-1",
"type": "SYSLOG",
"host": "10.71.140.32",
"protocol": "udp",
"additionalParameters": {
"type": "connector",
"properties": {
"property": [
{
"key": "remote.management.listener.port",
"value": "9001"
}
```

```
]
}
},
"port": 514
}
]
```


Retrieving Details of a Destination

This method displays details of the selected destination.

API Reference

```
GET /config/destinations/destination/<destination-name>
```

Sample Request

URI: `GET https://127.0.0.1:8443/config/destinations/destination/syslog-tcp-connector-1`

Sample Response

Status: 200 (OK)

Body:


```
{
"name": "syslog-tcp-connector-1",
"type": "SYSLOG",
"host": "10.71.140.31",
"protocol": "tcp",
"additionalParameters": {
"type": "connector",
"properties": {
"property": [
{
"key": "remote.management.listener.port",
"value": "9001"
}
```

```
]
}
},
"port": 514
}
```

Error Code

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Destination not found, destination=[syslog-tcp-connector-2]"
}
]
```

Reason: This error occurs when you try to retrieve a destination that is not present.

Creating a Destination

This method adds a new destination.

API Reference

```
POST /config/destinations/destination
```

Content-Type

application/json

Sample Request

URI: `POST https://127.0.0.1:8443/config/destinations/destination`

Body:

```
{
"name": "syslog-udp-connector-1",
"type": "SYSLOG",
```

```
"host": "10.71.140.32",
"protocol": "udp",
"additionalParameters": {
"type": "connector",
"properties": {
"property": [
{
"key": "remote.management.listener.port",
"value": "9001"
}
]
}
},
"port": 514
}
```


Sample Response

Status: 201 (Created)

Body:


```
{
"name": "syslog-udp-connector-1",
"type": "SYSLOG",
"host": "10.71.140.32",
"protocol": "udp",
"additionalParameters": {
"type": "connector",
"properties": {
"property": [
{
"key": "remote.management.listener.port",
"value": "9001"
}
```

```
]
}
},
"port": 514
}
```

Error Codes

Status: 400 (Bad Request)

Body:

```
[ { "errorSource": "Configuration", "description": "Duplicate name found for
Destination=[syslog-tcp-connector-1]" } ]
```

Reason: This error occurs when an existing destination is already using the name of the destination in the request.

Status: 400 (Bad Request)

Body:

```
[ { "errorSource": "Configuration", "description": "Duplicate hostname, port
and protocol found for Destination=[syslog-tcp-connector-3]: Hostname=
[10.71.140.33], Port=[514]" } ]
```

Reason: This error occurs when the combination of hostname, port, and protocol used in a new request is similar to an existing destination.

Deleting a Destination

This method deletes the selected destination.

API Reference

```
DELETE /config/destinations/destination/<name of the destination to be
deleted>
```

Sample Request

URI: `DELETE https://127.0.0.1:8443//config/destinations/destination/syslog-udp-connector-1`

Sample Response

Status: 200 (OK)

Body:

```
{ "name": "syslog-udp-connector-1", "type": "SYSLOG", "host": "10.71.140.32",
"protocol": "udp", "additionalParameters": { "type": "connector",
"properties": { "property": [ { "key": "remote.management.listener.port",
"value": "9001" } ] } }, "port": 514 }
```

Error Codes

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Destination not found, destination=[syslog-udp-connector-1]"
}
]
```

Reason: This error occurs when the destination that you are trying to delete is not present.

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "The destination=[SYSLOG-TCP-CONNECTOR-1] is being referenced
by one or more Destination Pools=[[syslog-tcp-connectors]]."
}
]
```

Reason: This error occurs when the destination you are trying to delete is present in a destination pool.

Retrieving a List of Destination Pools

This method displays all the destination pools.

API Reference

```
GET /config/destinationpools
```

Sample Request

URI: `GET https://127.0.0.1:8443/config/destinationpools`

Sample Response

Success Code: 200 (OK)

Body:

```
[
{
"name": "syslog-tcp-connectors",
"destinations": "syslog-tcp-connector-1,syslog-tcp-connector-2"
},
{
"name": "syslog-tcp-connectors2",
"destinations": "syslog-tcp-connector-1"
}
]
```

Retrieving Details of a Destination Pool

This method displays details of the selected destination.

API Reference

`GET /config/destinationpools/destinationpool/<name of the destination pool>`

Sample Request

URI: `GET https://127.0.0.1:8443//config/destinationpools/destinationpool/tcp-syslog-connectors`

Sample Response

Status: 200 (OK)

Body:

```
{
"name": "syslog-tcp-connectors",
```

```
"destinations": "syslog-tcp-connector-1"
}
```

Error Code

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Destination pool not found, destination pool=[tcp-syslog-connectors-non-existent]"
}
]
```

Reason: This error occurs when you try to retrieve a destination pool that is not present.

Creating a Destination Pool

This method adds a new destination pool.

API Reference

```
POST /config/destinationpools/destinationpool
```

Content-Type

application/json

Sample Request

URI: `POST https://127.0.0.1:8443/config/destinationpools/destinationpool`

Body:

```
{
"name": "syslog-udp-connectors",
"destinations": "syslog-udp-connector-1"
}
```

Sample Response

Status: 201 (Created)

Body:

```
{
"name": "syslog-udp-connectors",
"destinations": "syslog-udp-connector-1"
}
```

Error Codes

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Duplicate name found for Destination Pool=[syslog-tcp-connectors]"
}
]
```

Reason: This error occurs when the name of the destination pool you are trying to create is already present for some other destination pool.

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Undefined or invalid destination found for Destination Pool=[syslog-udp-connectors]: Destination=[syslog-udp-connector-1]"
},
{
"errorSource": "Configuration",
```

```
"description": "Found undefined Destination=[syslog-udp-connector-1]"
}
]
```

Reason: This error occurs when the destination mentioned in the request is not present.

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Destinations in destination pool should be of the same protocol: Destination Pool=[syslog-udp-connectors]"
}
]
```

Reason: This error occurs when the destinations' protocols present in the request are not same.

Deleting a Destination Pool

This method deletes the selected destination pool.

API Reference

```
DELETE /config/destinationpools/destinationpool/<name of the destination pool to be deleted>
```

Sample Request

URI: DELETE
`https://127.0.0.1:8443/config/destinationpools/destinationpool/syslog-udp-connectors`

Sample Response

Status: 200 (OK)

Body:

```
{
"name": "syslog-udp-connectors",
"destinations": "syslog-udp-connector-1"
```

```
}
```

Error Codes

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Destination pool not found, destination pool=[syslog-udp-connectors1]"
}
]
```

Reason: This error occurs when you try to delete a destination pool that is not present.

Adding a Destination to a Destination Pool

This method adds a destination to a destination pool.

API Reference

```
PUT /config/destinationpools/destinationpool/<name of the destination pool>/add/<name of the destination to be deleted>
```

Sample Request

URI: PUT

```
https://127.0.0.1:8443/config/destinationpools/destinationpool/syslog-tcp-connectors/add/syslog-tcp-connector-2
```

Sample Response

Status: 200 (OK)

Body:

```
{
"name": "syslog-tcp-connectors",
"destinations": "syslog-tcp-connector-1,syslog-tcp-connector-2"
}
```

Error Codes

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Duplicate destination found from the destination pool: destination pool=[syslog-tcp-connectors], destination=[syslog-tcp-connector-1]"
}
]
```

Reason: This error occurs when the destination pool already contains the destination you are trying to insert.

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Destinations in destination pool must be of the same protocol: destination pool=[syslog-tcp-connectors], destination=[syslog-udp-connector-1], expected protocol=[tcp]"
}
]
```

Reason: This error occurs when the destination you are trying to insert uses a different protocol than that of the existing destination in the same destination pool.

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
```

```
"description": "No such destination pool found: destination pool=
[destination-pool-1]"
}
]
```

Reason: This error occurs when the destination pool is not present.

Deleting a Destination From a Destination Pool

This method deletes a destination in a destination pool.

API Reference

```
DELETE /config/destinationpools/destinationpool/<name of the destination
pool>/delete/<name of the destination to be deleted>
```

Sample Request

URI: DELETE
`https://127.0.0.1:8443/config/destinationpools/destinationpool/syslog-tcp-connectors/delete/syslog-tcp-connector-1`

Sample Response

Status: 200 (OK)

Body:

```
{
"name": "syslog-tcp-connectors",
"destinations": ""
}
```

Error Codes

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
```

```
"description": "No such destination pool found: destination pool=
[destination-pool-1]"
}
]
```

Reason: This error occurs when the destination pool is not present.

Status: 400 (Bad Request)

Body:

```
[
{
"errorSource": "Configuration",
"description": "Configuration is not synchronized: destination pool=[syslog-
tcp-connectors], destination=[destination-1]"
}
]
```

Reason: This error occurs when the destination is not present.

REST API Common Errors

Following are the REST API common errors for Load Balancer:

No.	Error Messages
1	• Status: 401 (Unauthorized) • Body: `[{"errorSource":"Configuration","description":"Operation not supported for non-active primary node"}]` • Description: This error occurs when you call REST API using the IP address of the node that was not active during the time of the call.
2	• Status: 405 (Method not allowed) • Body: NA • Description: This error occurs when you do not pass the expected method type.

No.	Error Messages
3	• Status: 400 (Bad Request) • Body: `Unexpected character ('"' (code 34)): was expecting comma to separate OBJECT entries at [Source:org.glassfish.jersey.message.internal.ReaderInterceptorExecutor$UnCloseableInputStream@1a27bdb2; line: 9, column: 10] (through reference chain: com.arcsight.lb.bean.RoutingRule["additionalParameters"])` • Description: This error occurs when a comma is missing between two fields in the JSON object of the request body.
4	• Status: 404 (Not Found) • Body: NA • Description: This error occurs when the API name present in the URI is invalid.
5	• Status: NA • Body: No response or could not get any response. • Description: This error occurs when the IP address or the hostname or the port number in the URI is not correct.

Chapter 4 — Load Balancer Troubleshooting

This chapter contains information about troubleshooting for common issues.

Load Calculators Not Initialized/Destination Monitoring Not Working

Issue: Load calculators are not initialized and destination monitoring is not working, but the certificate import is fine.

2015-07-02 12:16:39,266 [ERROR][com.arcsight.lb.b.b][intialize] - Please check the credentials for Connector tcp-syslog-connector-12 Error Message [; nested exception is:

java.net.ConnectException: Connection refused]

2015-07-02 12:16:39,266 [ERROR][com.arcsight.lb.b.c][initDestinationLoadCalculator] - Failed to initialize Load Calculator for the destination [tcp-syslog-connector-12]

Answer: This problem can be resolved by:

1. Verifying that connector can be managed remotely by checking the value of `remote.management.enabled` in the `agent.properties` file. This value should be set to `true`.
2. Verifying that the `destination/additionalParameters/properties/property@remote.management.listener.port` matches the value of `remote.management.listener.port`.

Destination Configured with SCP Protocol but File Delivery Fails

Issue: Load Balancer cannot successfully log into the destination host with SCP.

Answer: It may log an exception. There are several possible reason for this error:

- `knownHostsFile` is not specified in the `lbConfig.xml` configuration file.
- `knownHostsFile` is configured, but the host key for this specific host was not found.
- `knownHostsFile` is configured and the host key was found, but the algorithm generated for the host key is neither RSA nor DSA. Currently Load Balancer supports only these two types of algorithms. If the host is configured to use another algorithm in generating a host key, regenerate the host key using one of the accepted algorithms.

Sources Relocated Away from [x] of [y] Destinations in Routing Rule

Issue: “Sources were reallocated away from [x] of [y] destinations in the routing rule [my-routing-rule]. You may wish to add more destinations” displays even though the "There was no incoming data" error message is being displayed.

Answer: The destination overloaded message is triggered by examining the Connector's internal statistics, regardless of the traffic that Load Balancer is sending it. It is therefore possible for a connector to be deemed overloaded even though Load Balancer has not yet sent it any traffic.

Calculating Loads for Routing

Issue: The metrics used to calculate SmartConnector loads do not represent the actual load of the SmartConnectors, and result in incorrect distribution of events.

Answer: Use a custom expression to set custom load-level calculation expressions for Weighted Round Robin and Aggregation Preferred routing policies, both as a global default and as per-destination overrides.

- For all destinations (excluding those which do not have their own expression,) configure `load.expression.default` in the `globalParameters` block.
- For only a specific destination, configure `load.expression` in the `additionalParameters` block. This overrides `load.expression.default`. Per-destination expressions can be used to favor certain destinations over others. Weaker destinations can be pre-favored to be less utilized by having a large constant value added or multiplied to their load.
- If neither are provided, the existing behavior is used.

Note: To specify a non-integer constant value (for example, 1.5), division must be used (for example, 3/2), as using a "." period is prohibited. Otherwise, normal operators and precedence rules apply. Higher return values indicate higher loads. There is no maximum value; values are automatically scaled relative to other load values. Negative values should not be used.

The following case-sensitive variables are available:

Variable Name	Data Type	Description
`eps`	float	Average events per second over a one-minute period.
`queueRate`	float	Queue rate over a one-minute period, if file queues are enabled.
`queueDropsTotal`	float	Total queue drops since the Connector started, if file queues are enabled.
`queueDrops`	float	Queue drops over a one-minute period, if file queues are enabled. This is calculated locally at the Load Balancer, and may erroneously read 0 if the Connector returns the same counter snapshot multiple times.
`cpuLoad`	int	Only available for Linux Connectors. The aggregate percentage of CPU time that was not idle, over a one-minute period, as an integer 0-100. (On non-Linux connectors, this will be -1.) *See further notes below.
`memUsed`	int	The instantaneous amount of memory (in megabytes) used by the heap when statistics were collected. This includes objects eligible for garbage collection, so may be significantly higher than is actually the case.
`memTotal`	int	The maximum size of the heap (in megabytes).

*About the `cpuLoad` variable:
1) This variable is only available for connectors running on Linux. It is only updated every 60 seconds by default, regardless of how often Load Balancer is polling. This is the average level for the entire period of time since the last polling interval. This value may be underreported for virtualized connectors or connectors with heavy disk traffic.

2) This variable includes all CPUs (and cores, and hardware threads) added together. For example, if the connector machine has a total of 4 hardware threads (1 socket, 2 cores, with 2 hardware threads per core), and only one hardware thread is at 100% usage while the other 3 are at 0% usage, this will be reported as 25% load. There is no way to distinguish this from all 4 hardware threads being at 25% load.

3) To use Linux destinations based solely on CPU load, unless they are dropping events from their queue, use: `cpuLoad + queueDrops * 100`

Appendix A –Configuration File Templates with Callout Information

Configuration templates exist for standalone and HA configurations. See the callouts within the files for additional information.

Standalone Mode Configuration Template File

The template files shown in this chapter are used to configure standalone and HA modes for Load Balancer.

```
<?xml version="1.0" encoding="UTF-8"?>
<lbConfiguration>
    <!-- Identify the current host among the memberHosts. -->
    <memberIdentity>primary-node</memberIdentity>

    <!-- Load Balancer can run in standalone mode.                                    --
>
    <!-- To run Load Balancer in standalone mode, configure one memberHost. vipAddress and      --
>
    <!-- vipPingPort cannot be null but it won't be referenced.                       --
>
    <memberHosts vipAddress="192.168.1.253" vipPingPort="9090">
        <!-- 'host' is the host address where Load Balancer is installed and 'port' is internally    --
>
        <!-- used to communicate with another Load Balancer to detect the health for HA support.     --
>
        <!-- Standalone mode still requires valid port number to be specified.            --
```

```
>
        <!-- If Load Balancer is running as non-root, add Load Balancer user to sudoer list and      --
>
        <!-- prefix 'vipBindCommand' and 'vipUnbindCommand' with 'sudo' such as 'sudo /sbin/ifup..'. --
>
        <memberHost name="primary-node" host="192.168.1.253" port="6702" isPrimary="true"
vipBindCommand="/sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="/sbin/ifdown
/etc/sysconfig/network-scripts/ifcfg-eth0:1"/>
    </memberHosts>
    <!-- To get an email notification when the Load Balancer member host is down or up, or when      --
>
    <!-- the destination is down or up, set enable to 'true' and configure the email section.        --
>
    <notification enable="true">
        <enabledNotification>
            <event name="MemberHostUp" message="Member node is up." />
            <event name="MemberHostDown" message="Member node is down." />
            <event name="DestinationUp" message="Destination is up." />
            <event name="DestinationDown" message="Destination is down." />
        </enabledNotification>
        <email>
            <!-- Create a prefix for the subject line.        -->
            <prefix>[Load Balancer]</prefix>
            <!-- Separate multiple recipients with a space.   -->
            <recipients>jane.doe@abc.com john.doe@abc.com</recipients>
            <sender>admin@abc.com</sender>
            <smtpServer>smtp.abc.com</smtpServer>
        </email>
    </notification>
```

```
    <routing>
        <!-- All names in the routing section must be unique. -->
        <destinationPools>
            <destinationPool name="tcp-syslog-connectors" destinations="syslog-connector-1,syslog-
connector-2"/>
            <destinationPool name="udp-syslog-connectors" destinations="syslog-connector-3,syslog-
connector-4"/>
            <destinationPool name="tls-syslog-connectors" destinations="syslog-connector-5,syslog-
connector-6"/>
            <destinationPool name="file-connectors" destinations="file-connector-1,file-connector-2"/>
        </destinationPools>
        <destinations>
            <!-- Examples of configuring TCP connectors as destinations. -->
            <!-- 'host' is the host address where the tcp connector is running and 'port' is
connector's listening   -->
            <!-- port which can be found from agent.properties. 'tcp' corresponds to 'Raw TCP' in
agent.properties.  -->
            <destination name="syslog-connector-1" type="syslog" host="192.168.1.12" port="513"
protocol="tcp">
                <!-- Specify the connection configuration value here with key and matching value. Load
Balancer need -->
                <!-- the information to perform the connector health check and to obtain the load
information.       -->
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                        <!-- Prefer to send fewer events to this Connector by counting its CPU load as
twice as busy. -->
```

```
                        <property key="load.expression" value="cpuLoad * 2"/>
                    </properties>
                </additionalParameters>
            </destination>
            <destination name="syslog-connector-2" type="syslog" host="192.168.1.13" port="513"
protocol="tcp">
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                    </properties>
                </additionalParameters>
            </destination>
            <!-- Examples of configuring UDP connectors as destinations. -->
            <destination name="syslog-connector-3" type="syslog" host="192.168.1.12" port="514"
protocol="udp">
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                    </properties>
                </additionalParameters>
            </destination>
            <destination name="syslog-connector-4" type="syslog" host="192.168.1.13" port="514"
protocol="udp">
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
```

```
                    </properties>
                </additionalParameters>
            </destination>
            <!-- Examples of configuring TLS connectors as destinations. -->
            <!-- As long as the connector is using the same certificate for the TLS syslog transport as
it is for -->
            <!-- remote management, Load Balancer will automatically work with it. -->
            <destination name="syslog-connector-5" type="syslog" host="192.168.1.12" port="515"
protocol="tls">
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                    </properties>
                </additionalParameters>
            </destination>
            <destination name="syslog-connector-6" type="syslog" host="192.168.1.13" port="515"
protocol="tls">
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                    </properties>
                </additionalParameters>
            </destination>
            <!-- Examples of configuring file-based connectors as destination.
            -->
            <!-- Supported protocols on the destination side are ftp and scp. Each protocol requires a
different    -->
```

```
            <!-- set of configuration values as shown in the examples below. Plaintext passwords are
persisted as   -->
            <!-- encrypted value when Load Balancer starts.
            -->
            <!-- In order to use 'scp' protocol, file that has ssh host key should be provided to
'knownHostsFile'. -->
            <!-- Load Balancer does not populate this file automatically. In order to obtain the host
key, ssh to   -->
            <!-- the destination manually and specify the full path of the system default known_hosts
file or copy  -->
            <!-- the file to another location and give that path to 'knownHostsFile'.
            -->
            <destination name="file-connector-1" type="file" path="/opt/connector-1/input"
host="192.168.0.1" protocol="scp" username="admin" password="password"
knownHostsFile="/root/.ssh/known_hosts">
                <!-- Configure the information about this connector before starting Load Balancer.
-->
                <additionalParameters type="connector">
                    <properties>
                        <!-- If the destination connector is file-based connector, specify the
following two    -->
                        <!-- values so that Load Balancer can handshake with the connector.
        -->
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                        <property key="agent.name" value="{agent name configured for the destination in
the final stage of agent setup}"/>
                    </properties>
                </additionalParameters>
```

```
            </destination>
            <!-- Configure the port if FTP server is configured with a non-default port. Default port
21 is used if not specified. -->
            <!-- 'host' is the host address of FTP server and 'username' and 'password' is the user
credential who has access to   -->
            <!-- FTP server. Plaintext password is encrypted and persisted to this file as soon as Load
Balancer starts.           -->
            <destination name="file-connector-2" type="file" host="192.168.0.2" protocol="ftp"
username="admin" password="password" path="landing">
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                        <property key="agent.name" value="{agent name configured for the destination in
the final stage of agent setup}"/>
                    </properties>
                </additionalParameters>
            </destination>
        </destinations>
        <routingRules>
            <!-- Supported routing policies are RoundRobin, WeightedRoundRobin, and
AggregationPreferred. -->
            <routingRule name="syslog-tcp-rule" sourceName="syslog-tcp" destinationPoolName="tcp-
syslog-connectors" routingPolicy="RoundRobin" enabled="true">
                <additionalParameters type="listener">
                    <properties>
                        <!-- Scan incoming messages to determine if there is a hostname or address
present, and insert one if not. -->
                        <property key="syslog.address.prepend.mode" value="scan"/>
```

```
                    </properties>
                </additionalParameters>
            </routingRule>
            <routingRule name="syslog-udp-rule" sourceName="syslog-udp" destinationPoolName="udp-
syslog-connectors" routingPolicy="WeightedRoundRobin" enabled="true"/>
            <routingRule name="syslog-tls-rule" sourceName="syslog-tls" destinationPoolName="tls-
syslog-connectors" routingPolicy="AggregationPreferred" enabled="true"/>
            <routingRule name="file-routing-rule" sourceName="file-watcher" destinationPoolName="file-
connectors" routingPolicy="RoundRobin" enabled="true"/>
        </routingRules>
        <sources>
            <!-- When the source is syslog type, set 'type' to 'syslog' and configure the protocol
accordingly. -->
            <!-- Supported protocols are 'udp', 'tcp', and 'tls'. 'port' is the listening port on Load
Balancer -->
            <!-- and source syslog server should be configured to send the events to this port.
        -->
            <source name="syslog-tcp" type="syslog" port="513" protocol="tcp"/>
            <source name="syslog-udp" type="syslog" port="514" protocol="udp"/>
            <source name="syslog-tls" type="syslog" port="515" protocol="tls"/>
            <!-- 'file' type source can be used with the file based connector and Load Balancer
downloads       -->
            <!-- the files from FTP server and distribute them to the defined destinations in
destination pool. -->
            <!-- Supported protocol is 'ftp' and 'host' is the host address where FTP server is
running.        -->
            <!-- It assumes a default FTP port 21. If other port is configured, add 'port' attribute
port and   -->
            <!-- specify the port number. 'path' should a relative path to FTP root directory.
```

```
        -->
            <!-- Specify the credential for one who has a permission in accessing files from  FTP
server with   -->
            <!-- a full permission since files will need to moved to another directory or deleted after
        -->
            <!-- the file is downloaded. When 'moveToDirectory' is configured, the downloaded file will
be      -->
            <!-- to a specified directory or when the value is empty, file will be deleted afterwards.
        -->
            <!-- If the specified path is located under 'path', be sure to give the path  as hidden
directory   -->
            <!-- starting with '.'. Otherwise it will attempt to download the files from the directory
again    -->
            <!-- since it will recursively look for sub-directories. To disable recursive lookup, set
false to  -->
            <!-- 'recursive' attribute.
        -->
            <!-- User credential who has access to FTP server and path should be specified in
'username' and    -->
            <!-- 'password'. Plaintext password will be converted as encrypted value as soon as Load
Balancer   -->
            <!-- starts.
        -->
            <!-- 'localWorkDirectory' is where the file is temporarily kept before the file is sent to
one of   -->
            <!-- the destinations. This is required value and it assumes that the directory exists
already.     -->
            <source name="file-watcher" type="file" host="192.168.0.2" protocol="ftp"
path="landingzone" username="admin" password="password" fileFilter=".*log" moveToDirectory=".done"
```

```
recursive="true" passive="false" localWorkDirectory="/tmp" />
        </sources>
    </routing>
    <!-- logInterval is in milliseconds. -->
    <statisticsLogging logInterval="60000"/>
    <!-- WebServer must be configured for all nodes listed as member hosts in Load Balancer.          -
->
    <webServer httpsPort="8443"/>
    <!-- Uncomment and configure this section in order to customize the configuration.               -
->
    <!-- Refer to the configuration guide for the details.                                           -
->
    <!-- globalParameters>
        <properties>
            <property key="batch.buffer.size" value="102400" />
            <property key="load.expression.default" value="cpuLoad"/>
        </properties>
    </globalParameters -->
</lbConfiguration>
```

HA Mode Configuration Template File

```
<?xml version="1.0" encoding="UTF-8"?>
<lbConfiguration>
    <!-- Identify the current host among the memberHosts. -->
    <memberIdentity>primary-node</memberIdentity>

    <!-- Load Balancer can run in HA mode: two hosts can run primary-secondary or peer.           -->
    <!--  (1) To run Load Balancer as primary-secondary, configure two memberHosts and            -->
    <!--      for one of the hosts, and set isPrimary to 'true'.                                   -->
    <!--  (2) To run Load Balancer as peer, configure two memberHosts and set isPrimary to        -->
    <!--      'false' for both.                                                                    -->
    <!-- 'vipAddress' is the virtual IP addres that will be shared between two member hosts to     -->
    <!-- handle seamless failover of member host. 'vipPingPort' is internally used to check if    -->
    <!-- VIP address is still bound to one of the member hosts for continuous event collection.   -->
    <!-- Specify any unused port to 'vipPingPort'.                                                -->
    <memberHosts vipAddress="192.168.1.255" vipPingPort="9090">
        <!-- 'host' is the host address where Load Balancer is installed and 'port' is internally  -->
        <!-- used to communicate with another Load Balancer to detect the health for HA support.   -->
        <!-- If Load Balancer is running as non-root, add Load Balancer user to sudoer list and      --
>
        <!-- prefix 'vipBindCommand' and 'vipUnbindCommand' with 'sudo' such as 'sudo /sbin/ifup..'. --
>
        <memberHost name="primary-node" host="192.168.1.253" port="6702" isPrimary="true"
vipBindCommand="/sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="/sbin/ifdown
/etc/sysconfig/network-scripts/ifcfg-eth0:1"/>
        <memberHost name="secondary-node" host="192.168.1.254" port="6702" isPrimary="false"
vipBindCommand="/sbin/ifup /etc/sysconfig/network-scripts/ifcfg-eth0:1" vipUnbindCommand="/sbin/ifdown
```

```
/etc/sysconfig/network-scripts/ifcfg-eth0:1"/>
    </memberHosts>
    <!-- To get an email notification when the Load Balancer member host is down or up, or when   -->
    <!-- the destination is down or up, set enable to 'true' and configure the email section.     -->
    <notification enable="true">
        <enabledNotification>
            <event name="MemberHostUp" message="Member node is up." />
            <event name="MemberHostDown" message="Member node is down." />
            <event name="DestinationUp" message="Destination is up." />
            <event name="DestinationDown" message="Destination is down." />
        </enabledNotification>
        <email>
            <!-- Create a prefix for the subject line.        -->
            <prefix>[Load Balancer]</prefix>
            <!-- Separate multiple recipients with a space.   -->
            <recipients>jane.doe@abc.com john.doe@abc.com</recipients>
            <sender>admin@abc.com</sender>
            <smtpServer>smtp.abc.com</smtpServer>
        </email>
    </notification>
    <routing>
        <!-- All names in the routing section must be unique. -->
        <destinationPools>
            <destinationPool name="tcp-syslog-connectors" destinations="syslog-connector-1,syslog-
connector-2"/>
            <destinationPool name="udp-syslog-connectors" destinations="syslog-connector-3,syslog-
connector-4"/>
            <destinationPool name="tls-syslog-connectors" destinations="syslog-connector-5,syslog-
connector-6"/>
```

```
            <destinationPool name="file-connectors" destinations="file-connector-1,file-connector-2"/>
        </destinationPools>
        <destinations>
            <!-- Examples of configuring TCP connectors as destinations. -->
            <!-- 'host' is the host address where the tcp connector is running and 'port' is
connector's listening   -->
            <!-- port which can be found from agent.properties. 'tcp' corresponds to 'Raw TCP' in
agent.properties.  -->
            <destination name="syslog-connector-1" type="syslog" host="192.168.1.12" port="513"
protocol="tcp">
                <!-- Specify the connection configuration value here with key and matching value. Load
Balancer need -->
                <!-- the information to perform the connector health check and to obtain the load
information.       -->
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                        <!-- Prefer to send fewer events to this Connector by counting its CPU load as
twice as busy. -->
                        <property key="load.expression" value="cpuLoad * 2"/>
                    </properties>
                </additionalParameters>
            </destination>
            <destination name="syslog-connector-2" type="syslog" host="192.168.1.13" port="513"
protocol="tcp">
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
```

```
{remote.management.listener.port from agent.properties}"/>
                    </properties>
                </additionalParameters>
            </destination>
            <!-- Examples of configuring UDP connectors as destinations. -->
            <destination name="syslog-connector-3" type="syslog" host="192.168.1.12" port="514"
protocol="udp">
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                    </properties>
                </additionalParameters>
            </destination>
            <destination name="syslog-connector-4" type="syslog" host="192.168.1.13" port="514"
protocol="udp">
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                    </properties>
                </additionalParameters>
            </destination>
            <!-- Examples of configuring TLS connectors as destinations. -->
            <!-- As long as the connector is using the same certificate for the TLS syslog transport as
it is for -->
            <!-- remote management, Load Balancer will automatically work with it. -->
            <destination name="syslog-connector-5" type="syslog" host="192.168.1.12" port="515"
protocol="tls">
```

```
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                    </properties>
                </additionalParameters>
            </destination>
            <destination name="syslog-connector-6" type="syslog" host="192.168.1.13" port="515"
protocol="tls">
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                    </properties>
                </additionalParameters>
            </destination>
            <!-- Examples of configuring file-based connectors as destination.
            -->
            <!-- Supported protocols on the destination side are ftp and scp. Each protocol requires a
different    -->
            <!-- set of configuration values as shown in the examples below. Plaintext passwords are
persisted as   -->
            <!-- encrypted value when Load Balancer starts.
            -->
            <!-- In order to use 'scp' protocol, file that has ssh host key should be provided to
'knownHostsFile'. -->
            <!-- Load Balancer does not populate this file automatically. In order to obtain the host
key, ssh to   -->
            <!-- the destination manually and specify the full path of the system default known_hosts
```

```
file or copy  -->
            <!-- the file to another location and give that path to 'knownHostsFile'.
            -->
            <destination name="file-connector-1" type="file" path="/opt/connector-1/input"
host="192.168.0.1" protocol="scp" username="admin" password="password"
knownHostsFile="/root/.ssh/known_hosts">
                <!-- Configure the information about this connector before starting Load Balancer.
-->
                <additionalParameters type="connector">
                    <properties>
                        <!-- If the destination connector is file-based connector, specify the
following two    -->
                        <!-- values so that Load Balancer can handshake with the connector.
        -->
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                        <property key="agent.name" value="{agent name configured for the destination in
the final stage of agent setup}"/>
                    </properties>
                </additionalParameters>
            </destination>
            <!-- Configure the port if FTP server is configured with a non-default port. Default port
21 is used if not specified. -->
            <!-- 'host' is the host address of FTP server and 'username' and 'password' is the user
credential who has access to   -->
            <!-- FTP server. Plaintext password is encrypted and persisted to this file as soon as Load
Balancer starts.           -->
            <destination name="file-connector-2" type="file" host="192.168.0.2" protocol="ftp"
username="admin" password="password" path="landing">
```

```
                <additionalParameters type="connector">
                    <properties>
                        <property key="remote.management.listener.port" value="
{remote.management.listener.port from agent.properties}"/>
                        <property key="agent.name" value="{agent name configured for the destination in
the final stage of agent setup}"/>
                    </properties>
                </additionalParameters>
            </destination>
        </destinations>
        <routingRules>
            <!-- Supported routing policies are RoundRobin, WeightedRoundRobin, and
AggregationPreferred. -->
            <routingRule name="syslog-tcp-rule" sourceName="syslog-tcp" destinationPoolName="tcp-
syslog-connectors" routingPolicy="RoundRobin" enabled="true">
                <additionalParameters type="listener">
                    <properties>
                        <!-- Scan incoming messages to determine if there is a hostname or address
present, and insert one if not. -->
                        <property key="syslog.address.prepend.mode" value="scan"/>
                    </properties>
                </additionalParameters>
            </routingRule>
            <routingRule name="syslog-udp-rule" sourceName="syslog-udp" destinationPoolName="udp-
syslog-connectors" routingPolicy="WeightedRoundRobin" enabled="true"/>
            <routingRule name="syslog-tls-rule" sourceName="syslog-tls" destinationPoolName="tls-
syslog-connectors" routingPolicy="AggregationPreferred" enabled="true"/>
            <routingRule name="file-routing-rule" sourceName="file-watcher" destinationPoolName="file-
connectors" routingPolicy="RoundRobin" enabled="true"/>
```

```
        </routingRules>
        <sources>
            <!-- When the source is syslog type, set 'type' to 'syslog' and configure the protocol
accordingly. -->
            <!-- Supported protocols are 'udp', 'tcp', and 'tls'. 'port' is the listening port on Load
Balancer -->
            <!-- and source syslog server should be configured to send the events to this port.
        -->
            <source name="syslog-tcp" type="syslog" port="513" protocol="tcp"/>
            <source name="syslog-udp" type="syslog" port="514" protocol="udp"/>
            <source name="syslog-tls" type="syslog" port="515" protocol="tls"/>
            <!-- 'file' type source can be used with the file based connector and Load Balancer
downloads       -->
            <!-- the files from FTP server and distribute them to the defined destinations in
destination pool. -->
            <!-- Supported protocol is 'ftp' and 'host' is the host address where FTP server is
running.        -->
            <!-- It assumes a default FTP port 21. If other port is configured, add 'port' attribute
port and   -->
            <!-- specify the port number. 'path' should a relative path to FTP root directory.
        -->
            <!-- Specify the credential for one who has a permission in accessing files from  FTP
server with   -->
            <!-- a full permission since files will need to moved to another directory or deleted after
        -->
            <!-- the file is downloaded. When 'moveToDirectory' is configured, the downloaded file will
be      -->
            <!-- to a specified directory or when the value is empty, file will be deleted afterwards.
        -->
```

```
            <!-- If the specified path is located under 'path', be sure to give the path  as hidden
directory   -->
            <!-- starting with '.'. Otherwise it will attempt to download the files from the directory
again    -->
            <!-- since it will recursively look for sub-directories. To disable recursive lookup, set
false to  -->
            <!-- 'recursive' attribute.
        -->
            <!-- User credential who has access to FTP server and path should be specified in
'username' and    -->
            <!-- 'password'. Plaintext password will be converted as encrypted value as soon as Load
Balancer   -->
            <!-- starts.
        -->
            <!-- 'localWorkDirectory' is where the file is temporarily kept before the file is sent to
one of   -->
            <!-- the destinations. This is required value and it assumes that the directory exists
already.     -->
            <source name="file-watcher" type="file" host="192.168.0.2" protocol="ftp"
path="landingzone" username="admin" password="password" fileFilter=".*log" moveToDirectory=".done"
recursive="true" passive="false" localWorkDirectory="/tmp" />
        </sources>
    </routing>
    <!-- logInterval is in milliseconds. -->
    <statisticsLogging logInterval="60000"/>
    <!-- WebServer must be configured for all nodes listed as member hosts in Load Balancer.          -
->
    <webServer httpsPort="8443"/>
    <!-- Uncomment and configure this section in order to customize the configuration.                -
```

```
->
    <!-- Refer to the configuration guide for the details.                                    -
->
    <!-- globalParameters>
        <properties>
            <property key="batch.buffer.size" value="102400" />
            <property key="load.expression.default" value="cpuLoad"/>
        </properties>
    </globalParameters -->
</lbConfiguration>
```

Send Documentation Feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this computer, click the link above and an email window opens with the following information in the subject line:

Feedback on Configuration Guide (Load Balancer 1.4.1)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to arcsight_doc@microfocus.com.

We appreciate your feedback!