

Extreme SLX-OS Management Configuration Guide, 20.1.1

Supporting ExtremeRouting and ExtremeSwitching
SLX 9640, SLX 9540, SLX 9250, and SLX 9150

9036287-00 Rev AA
February 2020



Copyright © 2020 Extreme Networks, Inc. All rights reserved.

Legal Notice

Extreme Networks, Inc. reserves the right to make changes in specifications and other information contained in this document and its website without prior notice. The reader should in all cases consult representatives of Extreme Networks to determine whether any such changes have been made.

The hardware, firmware, software or any specifications described or referred to in this document are subject to change without notice.

Trademarks

Extreme Networks and the Extreme Networks logo are trademarks or registered trademarks of Extreme Networks, Inc. in the United States and/or other countries.

All other names (including any product names) mentioned in this document are the property of their respective owners and may be trademarks or registered trademarks of their respective companies/owners.

For additional information on Extreme Networks trademarks, please see:

www.extremenetworks.com/company/legal/trademarks

Open Source Declarations

Some software files have been licensed under certain open source or third-party licenses. End-user license agreements and open source declarations can be found at:

www.extremenetworks.com/support/policies/software-licensing



Table of Contents

Preface.....	8
Text Conventions.....	8
Documentation and Training.....	10
Getting Help.....	10
Subscribe to Service Notifications.....	10
Providing Feedback.....	11
About This Document.....	12
Supported Hardware.....	12
Regarding Ethernet interfaces and chassis devices.....	12
Configuration Fundamentals.....	13
Configuration files.....	13
Default configuration files.....	13
Startup configuration files.....	14
Running configuration files.....	14
Displaying configurations.....	14
Backing up a running configuration.....	15
Backing up configurations.....	15
Configuration restoration.....	16
Managing flash files.....	16
Rebooting the device.....	18
Session connection.....	18
Telnet.....	19
SSH.....	21
Configuring the terminal session parameters.....	26
Configuring a login banner.....	27
Ethernet management interfaces.....	28
Displaying the management interface.....	28
Configuring an IPv6 address on the SLX platform.....	28
Port management.....	30
SLX 100G ports.....	30
Configuring breakout mode.....	31
10G/1G auto negotiation and auto detection mode.....	32
Port flap dampening.....	33
Port transition hold timer.....	34
Link fault signaling.....	35
Interface Ethernet ports.....	36
Displaying device interfaces.....	36
Interface reload delay to prevent traffic black-holing in vLAG.....	37
Scenario 1.....	37
Scenario 2.....	38

Configuration examples.....	39
Chassis and host names.....	40
Customizing chassis and host names.....	40
System clock.....	41
Setting the clock.....	41
Management VRFs.....	42
VRF reachability.....	42
Zero Touch Provisioning.....	45
Routing for ZTP.....	46
Using ZTP.....	46
ZTP configuration.....	47
Example of ZTP in a two-node topology	52
Enhanced Zero Touch Provisioning (ZTP+).....	55
Pre-requisites and Dependencies.....	56
ZTP+ Phases of Operation.....	56
Firmware validation.....	57
Configuration.....	58
MAC address aging.....	58
TCAM application-resource monitoring.....	58
TCAM library-resource monitoring.....	59
Hardware profiles.....	60
TCAM profiles.....	60
TCAM sharing.....	62
Counter profiles.....	63
FIB compression.....	64
Border profiles for Internet peering.....	65
Hardware profile show commands	68
Enter Maintenance Mode Before Performing Device Maintenance.....	68
SLX-OS and Linux Shell Interoperability.....	70
Overview	70
Limitations	70
Executing Linux shell commands from SLX-OS.....	71
Executing scripts from SLX-OS.....	72
Downloading a script to the SLX-OS device.....	72
Creating scripts in the Linux shell.....	72
Running scripts from the SLX-OS CLI.....	73
Accessing the Linux shell from SLX-OS.....	73
Executing SLX-OS commands from the Linux shell.....	73
Escalating Linux permissions to root.....	74
Saving and appending show command output to a file.....	75
Logs of Linux shell activities.....	75
Linux shell user entry and exit logs.....	75
Linux shell command execution logs.....	76
Configuring remote logging of Linux shell activities.....	77
Guest OS for TPVM.....	78
VM Access Management.....	78
Extreme SLX-OS VM Access Management.....	79
Insight Interface and TPVM.....	83

Insight interface port-channel.....	84
Insight interface.....	85
Inbound ACL-based mirroring.....	89
Insight interface traffic management and QoS.....	91
Configuring QoS egress scheduling.....	93
Troubleshooting port-mirroring.....	95
TPVM.....	97
Supported third-party applications, packages, and hardware.....	97
TPVM Installation and Management.....	99
Docker containers.....	111
Linux containers.....	113
Utilities installation and management.....	114
Assigning a static IP address on the TPVM Linux OS.....	116
TPVM on the SLX 9150 series.....	117
Network Time Protocol (NTP).....	118
Network Time Protocol overview.....	118
Date and time settings.....	118
Time zone settings.....	118
Network Time Protocol Server Overview.....	119
Network Time Protocol Client Overview.....	119
Network Time Protocol Associations.....	120
Network Time Protocol Authentication.....	120
Configuring NTP.....	122
Authenticating an NTP server.....	123
Displaying the active NTP server.....	124
NTP server status when an NTP server is not configured.....	124
NTP server status when an NTP server is configured.....	124
SNMP.....	125
SNMP overview.....	125
Basic SNMP operation.....	126
SNMP community strings.....	127
SNMP groups.....	127
SNMP users.....	127
SNMP views.....	128
SNMP server hosts.....	128
Multiple SNMP server context to VRF mapping.....	128
SNMP source interface.....	128
Configuring SNMPv2.....	129
Configuring SNMPv3.....	130
Configuring an SNMP server context to a VRF.....	131
Offline SNMP ifIndex generation tool.....	132
Generating ifIndexes for various interfaces.....	133
Configuration examples for generating ifIndexes offline.....	133
LLDP.....	135
LLDP overview.....	135
Layer 2 topology mapping.....	136
LLDP configuration guidelines and restrictions.....	137
Configuring and managing LLDP.....	137

Understanding the default LLDP.....	138
Disabling LLDP globally.....	138
Configuring LLDP global parameters.....	139
Configuring LLDP profiles.....	140
Configuring an LLDP profile to an interface.....	141
Displaying LLDP information.....	141
Clearing LLDP-related information.....	143
Account and Password Recovery.....	144
Recover the admin password from the root account.....	144
Root account and password recovery.....	144
(DNX devices) Recover the root login account.....	145
(XGS devices) Recover the root login account.....	146
(DNX devices) Recover the root password.....	146
(XGS devices) Recover the root password.....	147
Python Event-Management and Scripting.....	148
Python under Extreme operating systems	148
Python overview	148
Working interactively in the Python shell	149
Python scripts	150
Guidelines for writing Python scripts	150
Testing Python-script statements	151
Copying Python files to the device.....	151
Running Python scripts from the command line.....	153
Python scripts and run-logs	153
Python event-management	158
Configuring an event-handler profile	158
Activating an event-handler	159
Configuring event-handler options	159
Troubleshooting event-management.....	160
Aborting an event-handler action.....	160
Event-management show commands	160
Configuration Rollback.....	161
Configuration rollback overview.....	161
Supported topologies.....	161
Configuration rollback details.....	163
Configuration rollback considerations and limitations.....	164
General.....	164
Issues with specific configurations.....	165
RAS considerations.....	165
Intrusive scenarios.....	166
Performance considerations.....	166
Configuring rollback.....	166
Enabling or disabling rollback.....	166
Creating a default configuration checkpoint.....	166
Viewing checkpoint details.....	167
Modify the running configuration.....	167
Viewing the diff between a checkpoint and the running configuration.....	167
Viewing the patch between a checkpoint and the running configuration.....	168

Executing rollback.....	168
Verifying that the rollback diff is empty.....	168
Viewing rollback status.....	168
Viewing the rollback log.....	168
Viewing rollback log errors.....	169
Viewing current rollback status.....	169
Viewing rollback status history.....	170



Preface

This section describes the text conventions used in this document, where you can find additional information, and how you can provide feedback to us.

Text Conventions

Unless otherwise noted, information in this document applies to all supported environments for the products in question. Exceptions, like command keywords associated with a specific software version, are identified in the text.

When a feature, function, or operation pertains to a specific hardware product, the product name is used. When features, functions, and operations are the same across an entire product family, such as ExtremeSwitching switches or SLX routers, the product is referred to as *the switch* or *the router*.

Table 1: Notes and warnings




Icon	Notice type	Alerts you to...
	Tip	Helpful tips and notices for using the product.
	Note	Useful information or instructions.
	Important	Important features or instructions.

Table 1: Notes and warnings (continued)



Icon	Notice type	Alerts you to...
	Caution	Risk of personal injury, system damage, or loss of data.
	Warning	Risk of severe personal injury.

Table 2: Text

Convention	Description
<code>screen displays</code>	This typeface indicates command syntax, or represents information as it appears on the screen.
The words <i>enter</i> and <i>type</i>	When you see the word <i>enter</i> in this guide, you must type something, and then press the Return or Enter key. Do not press the Return or Enter key when an instruction simply says <i>type</i> .
Key names	Key names are written in boldface, for example Ctrl or Esc . If you must press two or more keys simultaneously, the key names are linked with a plus sign (+). Example: Press Ctrl+Alt+Del
<i>Words in italicized type</i>	Italics emphasize a point or denote new terms at the place where they are defined in the text. Italics are also used when referring to publication titles.
NEW!	This symbol identifies new content. In a PDF, this is searchable text.

Table 3: Command syntax

Convention	Description
bold text	Identifies command names, keywords, and command options.
<i>italic</i> text	Identifies a variable.
[]	Syntax components displayed within square brackets are optional. Default responses to system prompts are enclosed in square brackets.
{ x y z }	A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options.
x y	A vertical bar separates mutually exclusive elements.
< >	Nonprinting characters, such as passwords, are enclosed in angle brackets.
...	Repeat the previous element, for example, <i>member</i> [<i>member</i> ...].
\	Indicates a “soft” line break in command examples. If a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

Documentation and Training

Find Extreme Networks product information at the following locations:

[Current Product Documentation](#)

[Release Notes](#)

[Hardware/software compatibility matrices](#) for Campus and Edge products

[Supported transceivers and cables](#) for Data Center products

[Other resources](#), like white papers, data sheets, and case studies

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For details, visit www.extremenetworks.com/education/.

Getting Help

If you require assistance, contact Extreme Networks using one of the following methods:

Extreme Portal

Search the GTAC (Global Technical Assistance Center) knowledge base; manage support cases and service contracts; download software; and obtain product licensing, training, and certifications.

The Hub

A forum for Extreme Networks customers to connect with one another, answer questions, and share ideas and feedback. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.

Call GTAC

For immediate support: (800) 998 2408 (toll-free in U.S. and Canada) or 1 (408) 579 2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number, or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any actions already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

Subscribe to Service Notifications

You can subscribe to email notifications for product and software release announcements, Vulnerability Notices, and Service Notifications.

1. Go to www.extremenetworks.com/support/service-notification-form.
2. Complete the form (all fields are required).

3. Select the products for which you would like to receive notifications.

**Note**

You can modify your product selections or unsubscribe at any time.

4. Select **Submit**.

Providing Feedback

The Information Development team at Extreme Networks has made every effort to ensure the accuracy and completeness of this document. We are always striving to improve our documentation and help you work better, so we want to hear from you. We welcome all feedback, but we especially want to know about:

- Content errors, or confusing or conflicting information.
- Improvements that would help you find relevant information in the document.
- Broken links or usability issues.

If you would like to provide feedback, you can do so in three ways:

- In a web browser, select the feedback icon and complete the online feedback form.
- Access the feedback form at <https://www.extremenetworks.com/documentation-feedback/>.
- Email us at documentation@extremenetworks.com.

Provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.



About This Document

[Supported Hardware](#) on page 12

[Regarding Ethernet interfaces and chassis devices](#) on page 12

Supported Hardware

For instances in which a topic or part of a topic applies to some devices but not to others, the topic specifically identifies the devices.

SLX-OS 20.1.1 supports the following hardware platforms.

- Devices based on the Broadcom XGS[®] chipset family:
 - ExtremeSwitching SLX 9250
 - ExtremeSwitching SLX 9150
- Devices based on the Broadcom DNX[®] chipset family:
 - ExtremeRouting SLX 9640
 - ExtremeSwitching SLX 9540



Note

Although many software and hardware configurations are tested and supported for this release, documenting all possible configurations and scenarios is beyond the scope of this document.

For information about other releases, see the documentation for those releases.

Regarding Ethernet interfaces and chassis devices

The current SLX-OS version does not support any multi-slot (chassis) devices.

However, the Ethernet interface configuration and output *slot/port* examples in this document may appear as either *0/x* or *n/x*, where "n" and "x" are integers greater than 0.

For all currently supported devices, specify **0** for the slot number.



Configuration Fundamentals

- [Configuration files on page 13](#)
- [Session connection on page 18](#)
- [Ethernet management interfaces on page 28](#)
- [Configuring an IPv6 address on the SLX platform on page 28](#)
- [Port management on page 30](#)
- [Interface Ethernet ports on page 36](#)
- [Interface reload delay to prevent traffic black-holing in vLAG on page 37](#)
- [Chassis and host names on page 40](#)
- [System clock on page 41](#)
- [Management VRFs on page 42](#)
- [Zero Touch Provisioning on page 45](#)
- [Enhanced Zero Touch Provisioning \(ZTP+\) on page 55](#)
- [MAC address aging on page 58](#)
- [TCAM application-resource monitoring on page 58](#)
- [Hardware profiles on page 60](#)
- [Enter Maintenance Mode Before Performing Device Maintenance on page 68](#)

Configuration files

Extreme devices support three types of configuration files, default, startup, and running configuration files.

The startup configuration resides in the `/var/config/vcs/scripts` directory. Though the default configuration files are physically present in this directory, they are linked to the directory from their actual location.

When you boot up a device for the first time, the running configuration is identical to the startup configuration. As you configure the device, the changes are written to the running configuration. To save the changes, you must save the currently effective configuration (the running configuration) as the startup configuration. When the device reboots, the configuration changes become effective.

Default configuration files

Default configuration files are part of the firmware package for the device and are automatically applied to the startup configuration under the following conditions:

- When the device boots up for the first time and no customized configuration is available.

- When you restore the default configuration.

You cannot remove, rename, or change the default configuration. The default configuration file names are as follows:

- defaultconfig.standalone
- defaultconfig.cluster

Startup configuration files

The startup configuration is persistent. It is applied when the system reboots.

- When the device boots up for the first time, it uses the default configuration as the startup configuration, depending on the mode.
- When you make configuration changes to the running configuration and save the changes to the startup configuration with the **copy** command, the running configuration becomes the startup configuration.

The startup configuration file name is startup-config.

Running configuration files

The configuration currently effective on the device is referred to as the running configuration. Any configuration change you make while the device is online is made to the running configuration.

- The running configuration is nonpersistent.
- To save configuration changes, you must copy the running configuration to the startup configuration. If you are not sure about the changes, you can copy the changes to a file, and apply the changes later.

The running configuration file name is running-config.

Displaying configurations

The following examples illustrate how to display the default, startup, and running configurations, respectively.

Displaying the default configuration

To display the default configuration, enter the **show file** command with the default configuration filenames in privileged EXEC mode.

```
device# show file defaultconfig.standalone
device# show file defaultconfig.cluster
```

Displaying the startup configuration

To display the contents of the startup configuration, enter the **show startup-config** command in privileged EXEC mode.

```
device# show startup-config
```

Displaying the running configuration

To display the contents of the running configuration, enter the **show running-config** command in the privileged EXEC mode.

```
device# show running-config
```

Backing up a running configuration

Although configuration changes that you make are immediately included and saved in the running configuration, you should also save a backup of the running configuration.



Note

Before upgrading or downgrading the firmware, use one of the following tasks to backup the running configuration.

Applying previously saved configuration changes

When you are ready to apply the configuration changes you previously saved to a file, copy the file (*myconfig* in the example) to the startup configuration. The changes take effect after the device reboots.

Enter the **copy** command in privileged EXEC mode. Specify the file name as the file URL followed by the **startup-config** keyword.

```
device# copy flash://myconfig startup-config
This operation will modify your startup configuration. Do you want to continue? [Y/N]: y
```

Backing up configurations

Always keep a backup copy of your configuration files, so you can restore the configuration in the event the configuration is lost or you make unintentional changes.

The following recommendations apply:

- Upload the configuration backup copies to an external host or to an attached Extreme-branded USB device.
- Avoid copying configuration files from one device to another. Instead restore the device configuration files from the backup copy.

Copying a configuration file to an external host

You can copy the startup-config or running-config file to a remote server through FTP, SCP, TFTP, or SFTP.

In the following example, the startup configuration is copied to a file on a remote server by means of FTP.

```
device# copy startup-config ftp://admin:*****@10.34.98.133//archive/startup-config_device24-08_20101010
```

Backing up the startup configuration to a USB device

When you make a backup copy of a configuration file on an attached USB device, specify the USB and the destination file name on the USB device. You do not need to specify the target directory. The file is automatically recognized as a configuration file and stored in the default configuration directory.

1. Enable the USB device.

```
device# usb on
USB storage enabled
```

2. Enter the **copy startup-config** command with the destination (USB) and file name.

```
device# copy startup-config usb://startup-config_slx-08_20160510
```

Configuration restoration

Restoring a configuration involves overwriting a given configuration file on the device by downloading an archived backup copy from an external host or from an attached USB device.

All interfaces remain online. The following parameters are unaffected:

- Interface management IP address
- Software feature licenses installed on the device
- Virtual IP address

Restoring the default configuration

This restoration procedure resets the configuration to the factory defaults. The default configuration files are always present on the device and can be restored with the **copy** command.

To restore the default configuration, perform the following procedure in privileged EXEC mode.

1. Enter the **copy default-config startup-config** command to overwrite the running configuration with the default configuration.

```
device# copy default-config startup-config
```

2. Confirm that you want to make the change by entering Y when prompted.

```
This operation will modify your startup configuration. Do you want to continue? [Y/N]:
Y
```

3. Reboot the device.

```
device# reload system
```

Managing flash files

The Extreme device provides a set of tools for removing, renaming, and displaying files you create in the device flash memory. You can use the display commands with any file, including the system configuration files. The **rename** and **delete** commands only apply to copies of configuration files you create in the flash memory. You cannot rename or delete any of the system configuration files.

Listing the contents of the flash memory

To list the contents of the flash memory, enter the **dir** command in privileged EXEC mode.

```
device# dir
total 572
```



```
drwxr-xr-x 2 251 1011 4096 Jun 5 07:08 .
drwxr-xr-x 3 251 1011 4096 Mar 11 00:00 ..
-rw-r--r-- 1 root sys 410 Jun 3 00:56 defaultconfig.standalone
-rw-r--r-- 1 root sys 695 Jun 3 00:56 defaultconfig.cluster
-rw-r--r-- 1 root root 185650 Jun 5 09:38 startup-config
```

Deleting a file from the flash memory

To delete a file from the flash memory, enter the **delete** command with the file name in privileged EXEC mode.

```
device# delete myconfig
```



Note

You cannot delete a system configuration file in flash memory.

Renaming a flash memory file

To rename a file in the flash memory, enter the **rename** command with the source and destination file names in privileged EXEC mode.

```
device# rename myconfig myconfig_20101010
```



Note

You cannot rename a system configuration file in flash memory.

Viewing the contents of a file in the flash memory

To investigate the contents of a file in the flash memory, enter the **show file** command with the file name in privileged EXEC mode.

```
device# show file defaultconfig.cluster
vlan dot1q tag native
!
cee-map default
remap fabric-priority priority 0
remap lossless-priority priority 0
priority-group-table 15.0 pfc off
priority-group-table 1 weight 40 pfc on
priority-group-table 2 weight 60 pfc off
priority-table 2 2 2 1 2 2 2 15.0
!
!
port-profile default
vlan-profile
  switchport
  switchport mode trunk
  switchport trunk allowed vlan all
!
protocol lldp
!
!
logging auditlog class CONFIGURATION
logging auditlog class FIRMWARE
logging auditlog class SECURITY
```

```
!  
end
```

**Note**

To display the contents of the running configuration, use the **show running-config** command. To display the contents of the startup configuration, use the **show startup-config** command.

Rebooting the device

You can reboot the device with or without a power-on self-test (POST).

**Caution**

All reboot operations are disruptive, and the commands prompt for confirmation before executing. When you reboot a device, all traffic to and from it stops. All ports on that device remain inactive until the device comes back online.

**Note**

Any unsaved configurations are lost. During the boot process system initialization, configuration data (default or user-defined) are applied to the device through configuration replay.

- The **reload system** command performs a cold reboot that powers off and restarts the entire chassis. All session connections must be restarted. If the power-on self-test (POST) is enabled (via FIPS or CC enable), POST is executed when the system comes back up.

```
device# reload system
```

- The **fastboot** command reboots the device without a POST.

```
device# fastboot
```

Session connection

You can connect to your device through a console session on the serial port, or through a Telnet or Secure Shell (SSH) connection to the management port or the inband port belonging to either the mgmt-vrf, default-vrf, or a user-defined vrf. You can use any account login present in the local device database or on a configured authentication, authorization, and accounting (AAA) server for authentication. For initial setup procedures, use the pre-configured administrative account that is part of the default device configuration.

The device must be physically connected to the network. If the device network interface is not configured or the device has been disconnected from the network, use a console session on the serial port.

Refer to the appropriate hardware guide for information on connecting through the serial port and establishing an Ethernet connection for a console session.

Telnet

Telnet allows access to management functions on a remote networking device. Unlike SSH, Telnet does not provide a secure, encrypted connection to the device.

Telnet support is available in privileged EXEC mode on all Extreme platforms. The device supports a combined maximum (SSH, Telnet, and serial) of 32 non-root CLI sessions. Both IPv4 and IPv6 addresses are supported. Root users have another five dedicated sessions.

The Telnet service is enabled by default on the device. When the Telnet server is disabled, existing inbound Telnet connections are terminated and access to the device by additional inbound connections is not allowed until the Telnet server is re-enabled. If you have admin privileges, you can disable and re-enable inbound Telnet connections from global configuration mode.



Note

Outgoing Telnet connections from the device to any remote device are not affected by disabling or enabling the Telnet server in the device.



Note

When using Telnet, the root ID is blocked and you cannot login as root. Use **root enable** command to enable root ID.

Connecting to an Extreme device with Telnet

You can use the Telnet service to connect to the Extreme device.

A Telnet session allows you to access a device remotely using port 23. However, it is not secure. If you need a secure connection, use SSH.

1. Establish a Telnet session to the Extreme device from a remote device.

```
client# telnet 10.17.37.157
```

The example establishes a Telnet session to the device with the IP address of 10.17.37.157.

If the device is active and the Telnet service is enabled on it, a display similar to the following appears.

```
Trying 10.17.37.157...
Connected to 10.17.37.157.
Escape character is '^]'.

```

2. Once you have established the Telnet connection, you can log in normally.

```
device login: admin
Password:
SECURITY WARNING: The default password for at least
one default account (root, admin and user) have not been changed.

Welcome to the Extreme SLX-OS Software
admin connected from 10.252.24.5 using telnet on device
device#

```



Note

The default admin login name is admin. The default user name is user. The default password for both admin and user accounts is password.

Extreme recommends that you change the default account password when you log in for the first time. For more information on changing the default password, refer to the *Extreme SLX-OS Security Configuration Guide*.

Connecting to a remote device with Telnet

You can connect to a remote server from the device using a Telnet connection.

A Telnet session is not secure. If you need a secure connection, use SSH.

To connect to a remote server with Telnet, perform the following steps:

1. Establish a Telnet session connection to the remote device.

```
device# telnet 10.20.51.68 vrf mgmt-vrf
```

The example establishes a Telnet session to a device with the IP address of 10.20.51.68.

You can override the default port by using the **port-number** *port* option. However, the device must be listening on this port for the connection to succeed.

If the device is active and the Telnet service is enabled on it, a display similar to the following appears.

```
device# telnet 10.20.51.68 vrf mgmt-vrf
Trying 10.20.51.68...
Connected to 10.20.51.68.
Escape character is '^]'.
...
device login:
```

2. Once you have established the Telnet connection, you can log in normally.

Shutting down and re-enabling the Telnet service

The Telnet service is enabled by default. Shutting down the Telnet service forcibly disconnects all Telnet sessions running on a device.

To shut down and then re-enable the Telnet service, perform the following steps.

1. In privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. To shut down the Telnet service on the device, enter **telnet server shutdown**.

```
device(config)# telnet server use-vrf ?
Possible completions:
  <VRF Name> Provide the vrf (mgmt-vrf,default-vrf or <user defined vrf>) on which to
  start/stop telnet server
device# telnet server use-vrf red shutdown
```

All Telnet sessions including any currently active sessions are immediately terminated, and cannot be re-established until the service is re-enabled.

3. To re-enable the Telnet service on the device, enter **no telnet server shutdown**.

```
device(config)# no telnet server use-vrf red shutdown
```

**Note**

The **shutdown** option for a given VRF is displayed only when the Telnet server was configured and then shutdown on that VRF. Otherwise, the VRF name and shutdown option are not displayed for the **no** form of the command.

SSH

Secure Shell (SSH) allows secure access to management functions on a remote networking device. Unlike Telnet, which offers no security, SSH provides a secure, encrypted connection to the device.

SSH support is available in privileged EXEC mode on all Extreme platforms. The device supports a combined maximum (SSH, Telnet, and serial) of 32 non-root CLI sessions. Both IPv4 and IPv6 addresses are supported. Root users have another five dedicated sessions.

The SSH service is enabled by default on the device. When the SSH server is disabled, existing inbound SSH connections are terminated and access to the device by additional inbound connections are not allowed until the SSH server is re-enabled. If you have admin privileges, you can disable and re-enable inbound SSH connections from global configuration mode.

**Note**

Outgoing SSH connections from the device to any remote device are not affected by disabling or enabling the SSH server in the device.

**Note**

When using SSH, the root ID is blocked and you cannot login as root. Use **root enable** command to enable the root ID.

Feature support for SSH

SSHv2 is the supported version of SSH, but not all features typically available with SSHv2 are supported on the Extreme devices.

The following encryption algorithms are supported:

- **3des-cbc** Triple-DES
- **aes256-cbc**: AES in Cipher Block Chaining (CBC) mode with 256-bit key
- **aes192-cbc**: AES in CBC mode with 192-bit key
- **aes128-cbc**: AES in CBC mode with 128-bit key
- **aes256-gcm**: AES in Galios/Counter Mode (GCM) mode with 256-bit key
- **aes256-gcm@openssh.com**
- **aes192-gcm**: AES in GCM mode with 192-bit key
- **aes128-gcm**: AES in GCM mode with 128-bit key
- **aes128-gcm@openssh.com**
- **aes256-ctr**: AES in Counter Mode (CTR) mode with 256-bit key

- **aes192-ctr**: AES in CTR mode with 192-bit key
- **aes128-ctr**: AES in CTR mode with 128-bit key
- **blowfish-cbc**
- **cast128-cbc**
- **arcfour**
- **arcfour128**
- **arcfour256**
- **rijndael-cbc@lysator.liu.se**
- **chacha20-poly1305@openssh.com**

The following Hash-based Message Authentication Code (HMAC) message authentication algorithms are supported:

- **hmac-md5**: MD5 encryption algorithm with 128-bit key.
- **hmac-md5-96**
- **hmac-sha1**: SHA1 encryption algorithm with 160-bit key.
- **hmac-sha1-96**
- **hmac-sha2-256**: SHA2 encryption algorithm with 256-bit key.
- **hmac-sha2-256-etm@openssh.com**
- **hmac-sha2-512**: SHA2 encryption algorithm with 512-bit key.
- **hmac-sha2-512-etm@openssh.com**
- **hmac-ripemd160**
- **hmac-ripemd160@openssh.com**
- **umac-64@openssh.com**
- **umac-128@openssh.com**
- **hmac-sha1-etm@openssh.com**
- **hmac-sha1-96-etm@openssh.com**
- **hmac-md5-etm@openssh.com**
- **hmac-ripemd160-etm@openssh.com**
- **umac-64-etm@openssh.com**
- **umac-128-etm@openssh.com**
- **hmac-ripemd160-etm@openssh.com**

The following host keys are supported:

- ssh-dsa
- ssh-rsa
- ECDSA

The following key exchange algorithms are supported:

- diffie-hellman-group-exchange-sha256
- diffie-hellman-group-exchange-sha1
- diffie-hellman-group18-sha512

- diffie-hellman-group16-sha512
- diffie-hellman-group14-sha256
- diffie-hellman-group14-sha1
- diffie-hellman-group1-sha1
- curve25519-sha256
- curve25519-sha256@libssh.org
- ecdh-sha2-nistp256
- ecdh-sha2-nistp384
- ecdh-sha2-nistp521

SSH user authentication is performed with passwords stored on the device or on an external authentication, authorization, and accounting (AAA) server.

Connecting to an Extreme device with SSH

You can use SSH to connect to the Extreme device.

An SSH session allows you to access a device remotely using port 22.

1. Establish an SSH session connection to the Extreme device.

```
client# ssh admin@10.17.37.157
```

The example establishes an SSH session to the device with the IP address of 10.17.37.157.

2. Enter yes if prompted.

```
The authenticity of host '10.17.37.157 (10.17.37.157)' can't be established.  
RSA key fingerprint is 9f:83:62:cd:55:6c:b9:e8:1d:79:ab:b4:04:f4:f6:2a.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added '10.17.37.157' (RSA) to the list of known hosts.  
admin@10.17.37.157's password:
```

```
SECURITY WARNING: The default password for at least  
one default account (root, admin and user) have not been changed.
```

```
Welcome to the Extreme SLX-OS Software  
admin connected from 10.70.4.113 using ssh on device  
device#
```



Note

The default admin login name is admin. The default user login name is user. The default password for both admin and user accounts is password.

It is recommended that you change the default account password when you log in for the first time. For more information on changing the default password, refer to the *Extreme SLX-OS Security Configuration Guide*.

Connecting to a remote server with SSH

You can connect to a remote server from the device using the SSH (Secure Socket Handling) protocol to permit a secure (encrypted) connection.

To connect to a remote server with SSH, perform the following steps:

1. Establish an SSH connection with the login name and IP address for the remote server.

```
device# ssh 10.20.51.68 -l admin vrf mgmt-vrf
```

You can use the **-m** and **-c** options to override the default encryption and hash algorithms

2. Enter **yes** if prompted.

```
The authenticity of host '10.20.51.68 (10.20.51.68)' can't be established.  
RSA key fingerprint is ea:32:38:f7:76:b7:7d:23:dd:a7:25:99:e7:50:87:d0.  
Are you sure you want to continue connecting (yes/no)? yes
```

```
Warning: Permanently added '10.20.51.68' (RSA) to the list of known hosts.  
admin@10.20.51.68's password: *****
```

```
SECURITY WARNING: The default password for at least  
one default account (root, admin and user) have not been changed.  
Welcome to the Extreme SLX-OS Software  
admin connected from 10.20.51.66 using ssh on C60_68F
```


Managing SSH Client Public Keys

You can import SSH client public keys to establish an authenticated login to the device from an external ssh client.. You can also delete the key from the device to prevent it from being used for an authenticated login.

To manage the SSH client public keys, perform the following steps:

1. In privileged EXEC mode, import an SSH client public key to the device.

```
device# certutil import sshkey user admin host 10.70.4.106 directory /users/home40/
bmeenaks/.ssh file id_rsa.pub login fvt
```

This example imports the SSH client public key for the admin user from the remote 10.70.4.106 host using the directory and file information for the key and using the `fvt` login credentials for logging into the external server for the `scp`.

You can also copy the public key directly using **certutil sshkey user admin pubkey**. For example;

```
device# certutil sshkey user admin pubkey "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDnIm
+Ofjx/id3z2jDxXu9DcMuQqVq/NKi2Lms
+q7dA5Dqww8jlrOGawG8tMySOvnB1ZEvt1kqNneRi4l6Ot4/7hfd99rIOPGBP/NJs6xTLUrQhDgxB78ddTg
+6euBtkYLTAAaTC7kbXGXc08VVB9+4xrH+0bkvjU9RRvGJguUfdiFKEfIGVOyt0atdHi1dmgQ9BE0c065nc/
i9MjMJedBe174/QT4TxeGeEgaQ57c2AL5It2V4CzrZBDtnixdnHUO5w2vmBR61LZIDVT1fuX/
xYxDAm9H8SDpDX8pZ1fFpQBy/wrkIYPZ/p4OLrUApB/XAJGujrlN1ZLEu9U9MPVM/ root@ldap.hc-
fusion.in"
```

When the public key is imported (using **certutil import sshkey**) or copied (using **certutil sshkey**) for a user, password based authentication will become a fallback option for that particular user; This user will be allowed login using public key. If a user tries to login from any other machine for which public key is not present on the device then the user will be prompted for a password. Once the public key is removed for the user, only password based authentication will be enabled for that particular user.



Note

Whenever the public key is imported or removed, the SSH server is automatically rebooted and all active SSH connections are terminated.

2. Enter the password for the user.

```
Password: *****
```

When the SSH key is imported, the following message appears.

```
device# 2019/01/14-10:28:58, [SEC-3050], 75, INFO, SLX9540, Event: sshutil, Status:
success, Info: Imported SSH public key from 10.70.4.106 for user 'admin'.
```

3. Delete an SSH public key from the device to prevent it from being used. This resets the device to a password based login.

```
device# no certutil sshkey user admin
```

This example deletes the SSH client key for the admin user.



Note

Whenever the public key is imported or removed, the SSH server is automatically rebooted and all active SSH connections are terminated.

Shutting down and re-enabling the SSH service

The SSH service is enabled by default. Shutting down the SSH service forcibly disconnects all SSH sessions running on a device.



Note

When shutting down the service, either the SSH or the Telnet server on the Management VRF must remain operational. For example; if the Telnet server on the default-vrf and mgmt-vrf are shut down, the ssh server can be disabled on the default-vrf, but NOT on the mgmt-vrf.

To shut down and then re-enable the SSH service, perform the following steps.

1. In privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Shut down the SSH service on the device.

```
device(config)# ssh server use-vrf
Possible completions:
<VRF Name> Provide the vrf (mgmt-vrf,default-vrf or <user defined vrf>) on which to
start/stop ssh server
device(config)# ssh server use-vrf default-vrf shutdown
```

All SSH sessions on the specified vrf are immediately terminated, and cannot be re-established until the service is re-enabled.

3. To re-enable the SSH service on the device, enter **no ssh server shutdown**.

```
device(config)# no ssh server use-vrf default-vrf shutdown
```



Note

The shutdown option for a given VRF is displayed only when the SSH server was configured and then shutdown on that VRF. Otherwise, the VRF name and shutdown option are not displayed for the no form of the command.



Note

Additionally, the SSH Server may be restarted on all VRF instances using `ssh-server restart`.

Configuring the terminal session parameters

You can set the terminal parameters for the current session. You can also set password attributes for the length of the session and login attempts.

To set the parameters, perform the following steps:

1. In privileged EXEC mode, set the display length.

```
device# terminal length 30
```

This example sets the lines to be displayed on the terminal session at 30 lines.



Note

Setting the terminal length to 0 removes page breaks for the show commands' output.

2. Set the timeout length.

```
device# terminal timeout 3600
```

This example sets the timeout of 3600 seconds (60 minutes) for the terminal session.



Note

Specifying a value of 0 allows the terminal session to stay open until the device is rebooted or the connection is terminated by other means.

3. Access global configuration mode.

```
device# configure terminal
```

4. Configure the maximum login attempts to establish a session.

```
device(config)# password-attributes max-retry 4
```

This example sets the maximum login attempts of four to establish a session.

5. Set the maximum number of minutes the user account remains locked when user fails to login within the maximum login attempts.

```
device(config)# password-attributes admin-lockout max-lockout-duration 5
```

This example specifies that the user account be unlocked after 5 minutes.

The following configuration is the example of the previous steps.

```
device# terminal length 30
device# terminal timeout 3600
device# configure terminal
device(config)# password-attributes max-retry 4
device(config)# password-attributes admin-lockout max-lockout-duration 5
```

Configuring a login banner

The Extreme device can be configured to display a greeting message on user terminals as a banner when they enter the Privileged EXEC CLI level or access the device through Telnet/SSH.

Complete the following steps to set and display a banner.

1. In privileged EXEC mode, access global configuration mode.

```
device# configure terminal
Entering configuration mode terminal
```

2. Configure the login banner.

```
device(config)# banner login "Please do not disturb the setup on this device"
```

This example configures a text message on a single line by enclosing the text in double quotation marks (" ").

The banner can be up to 2048 characters long. To create a multi-line banner, enter the **banner login** command followed by the **Esc-m** keys. Enter **Ctrl-D** to terminate the input.

You can use the **no banner login** command to remove the banner.

3. Verify the configured banner.

```
device(config)# do show running-config banner
```

The configured banner is displayed.

```
banner login "Please do not disturb the setup on this device"
```

The following example is the configuration of the previous steps.

```
device# configure terminal
Entering configuration mode terminal
device(config)# banner login "Please do not disturb the setup on this device"
```

Ethernet management interfaces

The management Ethernet network interface provides management access, including direct access to the device CLI. You must configure at least one IP address using a serial connection to the CLI before you can manage the system. You can either configure static IP addresses, or you can use a Dynamic Host Configuration Protocol (DHCP) client to acquire IP addresses automatically. For IPv6 addresses, both static IPv6 and stateless IPv6 autoconfiguration are supported.



Important

Setting static IPv4 addresses and using DHCP are mutually exclusive. If DHCP is enabled, remove the DHCP client before you configure a static IPv4 address. However, this does not apply to IPv6 addresses.

Displaying the management interface

You can display the information about the management interface on the device. If an IP address has not been assigned to the network interface, you must connect to the CLI using a console session on the serial port. Otherwise, connect to the device through Telnet or SSH.

```
device# show interface management 0
interface management 0
  line-speed actual "1000baseT, Duplex: Full"
  line-speed configured Auto
  oper-status up
  ip address "static 10.20.161.66/20"
  ipv6 ipv6-address [ "static 2620:100:0:f814:10:20:161:66/64 preferred" ]
```

Configuring an IPv6 address on the SLX platform

Following are the basic pre-requisites for configuring an IPv6 address on the SLX platform:

- PC connected to the serial port of the device.
- IPv6 network assignment with a netmask and router address from the network administrators. This will generally be a /64 network.



Note

If you are provided an IPv6 prefix with a /65 to /128 net mask, assign the addresses according to your network administrator's direction, and do NOT follow this procedure.

To configure IPv6 addresses, perform the following steps:

1. Enter the show system command to know the STACK MAC and the BURNED IN MAC.

```
device# show system
Stack MAC           : 60:9c:9f:60:88:00

  -- UNIT 0 --
Unit Name           : 9450
Switch Status       :
Hardware Rev        :
Up Time             : up 21:00
Current Time        : 23:08:49 GMT
SLX-OS Version      : 18x.1.00
Jumbo Capable       : yes
Burned In MAC       : 60:9c:9f:46:e2:06
Management IP       : 10.25.101.4
Management Port Status : UP
```

The MAC addresses are used to create the IPv6 SLAAC address for the following mapping:

- Stack MAC - IPv6 address for chassis virtual-ipv6.
2. Convert each MAC address to a modified EUI-64 format, and then into the final IPv6 address for the interfaces by performing the following steps:

- a. Remove any punctuation from the MAC.

```
609c9f46e206
```

- b. Insert **fffe** after the first 6 characters.

```
609c9ffffe46e206
```

- c. Using a calculator application in HEX Mode on a PC, do a Bitwise OR operation of the modified MAC with 0200000000000000.

```
629c9ffffe46e206
```

- d. Convert the result to IPv6 format by inserting colons after every 4 characters from the right hand side.

```
629c:9fff:fe46:e206
```

- e. Prepare the IPv6 network information for use. This example uses a sample network of 2001:DB8::/32 provided by the Admin.

- Normalize the address to a fully expanded format.

```
2001:0DB8:0000:0000:0000:0000:0000:0000/32
```

- Remove the cidr notation.

```
2001:0DB8:0000:0000:0000:0000:0000:0000
```

- Remove the host portion of the address based on a /64 netmask.

```
2001:0DB8:0000:0000:
```

- Contract the remaining portion of the address of any leading zeros.

```
2001:DB8::
```

- f. Combine the IPv6 network prefix from step 2e and the result of step 2d to make the IPv6 address.

```
2001:DB8::629c:9fff:fe46:e206/32
```

- g. Repeat steps 2a to 2f for each MAC address.

3. Apply the addresses to the appropriate interfaces and configure the default route using the router address provided by the network administrator.

Port management

The Extreme device allows the port management of the following features for interface Ethernet ports.

- SLX 9540 port management includes the following:
 - Supports 54 ports in total. Ports 1 - 48 support 10G, 1G and 100 Mbps speed (default is 10G).
 - Ports 49-54 support 40G, 100G; and also support 4x10G and 4x25G breakout configurations. Default is 100G.
 - Forward Error Correction (FEC) is supported only in 100G mode.
- SLX 9640 port management includes the following:
 - Supports 36 ports in total. Ports 1 - 24 support 10G and 1G speed (default is 10G).
 - Ports 25-36 support 40G, 100G; and also support 4x10G, 4x25G, and 2x50G breakout configurations. Default is 100G.
 - Forward Error Correction (FEC) is supported only in 100G mode.
- SLX-9250 port management includes the following:
 - Supports 32 ports of 40G and 100G.
 - Ports may be broken out into 4x10G or 4x25G.
- SLX-9150-48Y port management includes the following:
 - Supports 56 ports in total
 - 48 ports support 1G, 10G, and 25G.
 - 8 ports support 40G, and 100G. These ports are able to break out to 4x10G.
 - 4x25Gb is supported on 2 ports only (0/49 and 0/56).
- 9150-48XT port management includes the following:
 - Supports 54 ports in total.
 - 48 ports support 1G and 10G.
 - 6 ports support 40G and 100G.
 - Ports 49 and 54 support break out configurations of 4x10G or 4x25G.
- Interface Ethernet port management features discussed this section include the following:
 - Port transition hold timer
 - Port flap dampening
 - Link fault signaling

SLX 100G ports

For fixed form factor SLX devices, all 100Gb/40Gb interfaces default to 100Gb mode.

You can configure 40G mode using the **speed 40000** command from the interface configuration mode. Each 100G port also supports 4x25G and 4x10G breakout configurations.

Configuring breakout mode

On the fixed form factor SLX, you can configure any 100/40G port as four 25G or 10G ports.

Before performing the following procedure, you can verify the current port configuration using the **show interface status** command:

Port	Status	Mode	Speed	Type	Description
Eth 0/1	adminDown	--	--	--	
Eth 0/2	adminDown	--	--	--	
Eth 0/3	adminDown	--	--	--	
Eth 0/4	adminDown	--	--	--	
Eth 0/5	adminDown	--	--	--	



Note

When configuring breakout mode - either breaking into multiple interfaces or consolidating into one interface - it is a best practice to remove all configuration on the interface, and set the interface to the disabled state.

Perform the following steps:

1. Access global configuration mode.

```
device# configure terminal
```

2. Shut down the port or ports to be configured.

```
device (config)# interface ethernet 0/1
shutdown
exit
```

Or;

```
device (config)# interface ethernet 0/1:1-4
shutdown
exit
```

3. Access hardware configuration mode.

```
device(config)# hardware
```

4. Access the port to be configured.

```
device(config-hardware)# connector 0/1
```

5. Set the breakout mode.

```
device(config-connector-0/1)# breakout mode 4x10g
```



Note

Dynamic breakout is supported; the user does not need to reboot the switch to execute the breakout.

6. Exit configuration mode.

```
device(config-connector-0/1)# exit
device(config-hardware)# exit
device(config)#
```

7. Verify the configuration.

```
device(config)# show interface status
```

```
-----
```

Port	Status	Mode	Speed	Type	Description
Eth 0/1:1	adminDown	--	--		
--					
Eth 0/1:2	adminDown	--	--		
--					
Eth 0/1:3	adminDown	--	--		
--					
Eth 0/1:4	adminDown	--	--		
--					
Eth 0/2	adminDown	--	--		
--					
Eth 0/3	adminDown	--	--		
--					
Eth 0/4	adminDown	--	--		
--					
Eth 0/5	adminDown	--	--	--	

10G/1G auto negotiation and auto detection mode

The SLX supports 10G/1G auto negotiation and auto detection mode.

- Auto negotiation is supported on ports 1 to 24 on the front plate. However, ports 25 to 72 support 1G mode without auto negotiation.
- Auto detection occurs when the interface speed is configured based on the detected optic type.
- Only full duplex is supported in the CL37 auto-negotiation.

You can manually configure the port speed. In manual mode, the inserted optic must match the configured speed. Otherwise, the link will not come up. You can configure 1G mode with or without auto negotiation. The following speed matrix shows different combinations of modes on the SLX 9540.

Table 4: Port speed matrix

	Ports 1 to 48 on SLX 9540
speed auto (default)	Auto-detection: <ul style="list-style-type: none"> • If 1G SFP optic is detected: Port is configured as 1G with 1000Base-X AN enabled. • If 1G SFP copper is detected: Port is configured as SGMII with 1000Base-T AN enabled; 1G, 100Mbps Full-Duplex advertised. • If 10G SFP is detected: port is configured as 10G
speed 1000	Force to 1G mode, AN disabled
speed 1000-auto	Force to 1G mode, AN enabled <ul style="list-style-type: none"> • If 1G SFP optic is detected: Port is configured as 1G with 1000Base-X AN enabled. • If 1G SFP copper is detected: Port is configured as SGMII with 1000Base-T AN enabled; 1G, 100Mbps Full-Duplex advertised.
speed 10000	Force to 10G mode

Table 4: Port speed matrix (continued)

	Ports 1 to 48 on SLX 9540
Speed 100	<ul style="list-style-type: none"> If 1G SFP optic is detected: No configuration change; link stays down If 1G SFP copper is detected: Port is configured as SGMII (AN enabled) but 1000Base-T AN disabled.
no speed	Equivalent to speed auto

Port flap dampening

Port flap dampening allows you to configure a wait period before a port, whose link goes down then up, becomes enabled.

If the port link state toggles, from down to up or from up to down, for a specified number of times within a specified period, the interface is physically disabled for the specified wait period. Once the wait period expires, the port's link state is re-enabled. However, if the wait period is set to zero (0) seconds, or you want to re-enable the port before the wait period expires, the port must be manually re-enabled.

Configuring port flap dampening

By default, port flap dampening is disabled on the device. You can configure the threshold of link flapping to shut down the port and the time interval in which it remains shut down. This feature is available for all front ports on the device and is configured on the interface level.

Perform the following steps to configure port flap dampening:

1. From privileged EXEC mode, access global configuration mode.

```
device# configure terminal
```

2. Access the interface configuration mode for the port that you want to configure.

```
device(config)# interface Ethernet 1/4
```

3. Configure port flap dampening.

```
device(conf-if-eth-1/4)# link-error-disable 10 3 10
```

In this example, the values for the parameters are as follows:

- The toggle threshold is set to 10 times. The threshold is the number of times that the port's link state goes from up to down and down to up before the wait period is activated.
- The sampling time is set to 3 seconds. This time period is the amount of time during which the specified toggle threshold can occur before the wait period is activated.
- The wait time is set to 10 seconds. This period of time is the amount of time the port remains disabled (down) before it becomes enabled. Entering 0 indicates that the port will stay down until an administrative override occurs.

The following example shows the steps in the previous configuration.

```
device# configure terminal
device(config)# interface Ethernet 1/4
device(conf-if-eth-1/4)# link-error-disable 10 3 10
```

Port transition hold timer

The port transition hold timer delays the sending of port up or down events to Layer 2 protocols and prevents port link flapping from affecting upper layer protocols or applications. After the delay expires, the event is sent to the upper layer.

While link down events are reported immediately in the Syslog, their effect on higher level protocols such as OSPF is delayed according to how the hold timer is configured. When configured, the timer affects the physical link events. However, the resulting logical link events are also delayed.



Note

All LAG member ports must have the same delayed-link-event configuration.



Note

The delayed-link-event configuration is applicable only on a physical interface. It is not valid on a VLAN, VE, LAG, or loopback interfaces.



Note

The port transition hold timer does not take effect when the interface is administratively shut down.

Configuring the port transition hold timer

By default, the sending of an up or down port event is not delayed. You can configure a delay for either or both events.

Perform the following steps to configure the port transition hold timer:

1. From privileged EXEC mode, access global configuration mode.

```
device# configure terminal
```

2. Access the interface configuration mode for the port that you want to configure.

```
device(config)# interface Ethernet 4/2
```

3. Configure the port transition hold timer.

```
device(conf-if-eth-4/2)# delay-link-event 2 down
```

The polling iteration is 50 ms. In this example, 50 ms is multiplied by 2 and the sending of port down event is delayed by 100 ms. If the port is detected to be in the up state within the 100 ms, the delayed down event is cancelled.

You can specify a multiplier value from 1 to 200 for delay times from 50 ms to 10 seconds and a port event of **up**, **down**, or **both**.

4. Verify the configuration.

```
device(conf-if-eth-4/2)# do show running-config internet ethernet 4/2
interface Ethernet 4/2
...
delay-link-event 2 down
no shutdown
!
```

The following example shows the steps in the previous configuration.

```
device# configure terminal
device(config)# interface Ethernet 4/2
device(conf-if-eth-4/2)# delay-link-event 2 down
```

Link fault signaling

The SLX platform supports Link Fault Signaling (LFS) detection for interface types of 10G, 40G, 100G, and 40G breakout ports. It detects local and remote faults.



Note

LFS is not supported in 1G mode.

When the device detects a local fault, it returns a remote fault to the link partner. When the device detects a remote fault, it returns an idle state.

A port's physical link detection is independent of LFS detection. When either of these link fault signals is detected, the following behaviors occur:

- The link is declared as DOWN and the port should display Protocol Down on the SLX-OS CLI.
- The physical link is not brought down in both of the previous cases. The peer side based on its implementation might display that the link is UP when the Extreme device displays that the link is DOWN due to a fault detection.
- The transmit (TX) packets, if any, are dropped at the MAC layer. The receive (RX) packets, if any, are dropped in the software.
- The detected signal is reported as a RASTRACE message on the line card. The same information is reported on the MM as a RASLOG. The same behavior occurs when the signal is cleared.

You can enable or disable LFS globally and on the interface level for both RX and TX directions:

- If the LFS is enabled for RX, the normal local and remote fault detection and processing described previously occur. If it is disabled for RX, local and remote fault detection are ignored.
- If the LFS is enabled for TX and a local fault occurs, a remote fault (pause frame) is generated to the remote side. If it is disabled for TX, the remote fault is not generated.

Configuring link fault signaling

By default, both TX and RX LFS are enabled. You can disable either RX or TX LFS globally or on the interface level.

Perform the following steps to configure LSF globally or on an interface.

1. From privileged EXEC mode, access global configuration mode.

```
device# configure terminal
```

2. Globally change the LFS, if required.

```
device(config)# link-fault-signaling rx off tx on
```

In this example, the global LFS is disabled for the link fault RX and enabled for link fault TX.

3. Access the interface configuration mode for the port that you want to configure.

```
device(config)# interface Ethernet 0/1
```

4. Shut down the interface.

```
device(conf-if-eth-0/1)# shutdown
```

The interface must be in the shutdown state before you disable or enable TX LFS.

5. Change the LFS for the interface.

```
device(conf-if-eth-0/1)# link-fault-signaling rx on tx off
```

In this example, the LFS for the interface is enabled for the link fault RX and disabled for the link fault TX. This configuration on the interface overrides the global configuration.

6. Enable the interface.

```
device(conf-if-eth-0/1)# no shutdown
```

7. Verify the configuration for the interface.

```
device(conf-if-eth-0/1)# do show running-config interface ethernet 0/1
interface Ethernet 0/1
...
  link-fault-signaling rx on tx off
  no shutdown
!
```

The following example shows the steps in the previous configuration.

```
device# configure terminal
device(config)# link-fault-signaling rx off tx on
device(config)# interface Ethernet 0/1
device(conf-if-eth-0/1)# shutdown
device(conf-if-eth-0/1)# link-fault-signaling rx on tx off
device(conf-if-eth-0/1)# no shutdown
```

Interface Ethernet ports

All Extreme device ports are pre-configured with default values that allow the device to be fully operational at initial startup without any additional configuration. In some configuration scenarios, changes to the port parameters may be necessary to adjust to attached devices or other network requirements.

Displaying device interfaces

The device supports Ethernet, loopback, management, and virtual Ethernet interfaces (VEs).

Enter the **show running-config interface** command to display the interfaces and their status.

The following example displays the Ethernet interfaces on the device and are identified by the port number.

For example, the notation 0/8 indicates port 8 on a device.

```
device# show running-config interface ethernet
interface Ethernet 0/1
  no shutdown
!
interface Ethernet 0/2
  channel-group 101 mode active type standard
  lacp timeout long
  no shutdown
!
interface Ethernet 0/3
  channel-group 101 mode active type standard
  lacp timeout short
  no shutdown
!
```

```
interface Ethernet 0/4
 shutdown
!
interface Ethernet 0/5
 shutdown
!
interface Ethernet 0/6
 shutdown
!
interface Ethernet 0/8
 channel-group 143 mode active type standard
 lacp timeout short
 no shutdown
!
interface Ethernet 0/9
 shutdown
!
```

Interface reload delay to prevent traffic black-holing in vLAG

The bring-up of edge interfaces before a vLAG is formed and before BGP routes converge results in traffic black-holing. The **reload-delay** feature addresses that problem by delaying the forming of Link Aggregation Control Protocol (LACP) on interfaces. The **reload-delay** command configures a global delay-time value for all interfaces on which reload delay is enabled but the delay is not specified.

The **reload-delay** feature has two configuration commands.

- `reload-delay <1-3600>` Global delay time (all unconfigured interfaces) configures a default delay-time value. This value is applied to the interfaces on which reload-delay is enabled but a delay-time value has not been configured.
- `reload-delay enable <1-3600>` Interface delay time; the interface configuration always takes precedence over the global configuration.

Consider the two following scenarios.

Scenario 1

Node1 in VCS Cluster1 is reloading. The vLAG between Node1 and Node2 is not formed yet, but the BGP session between leaf and spine nodes is established. Servers could start load balancing the traffic to Node1, but that traffic is black holed as the vLAG is not formed yet.

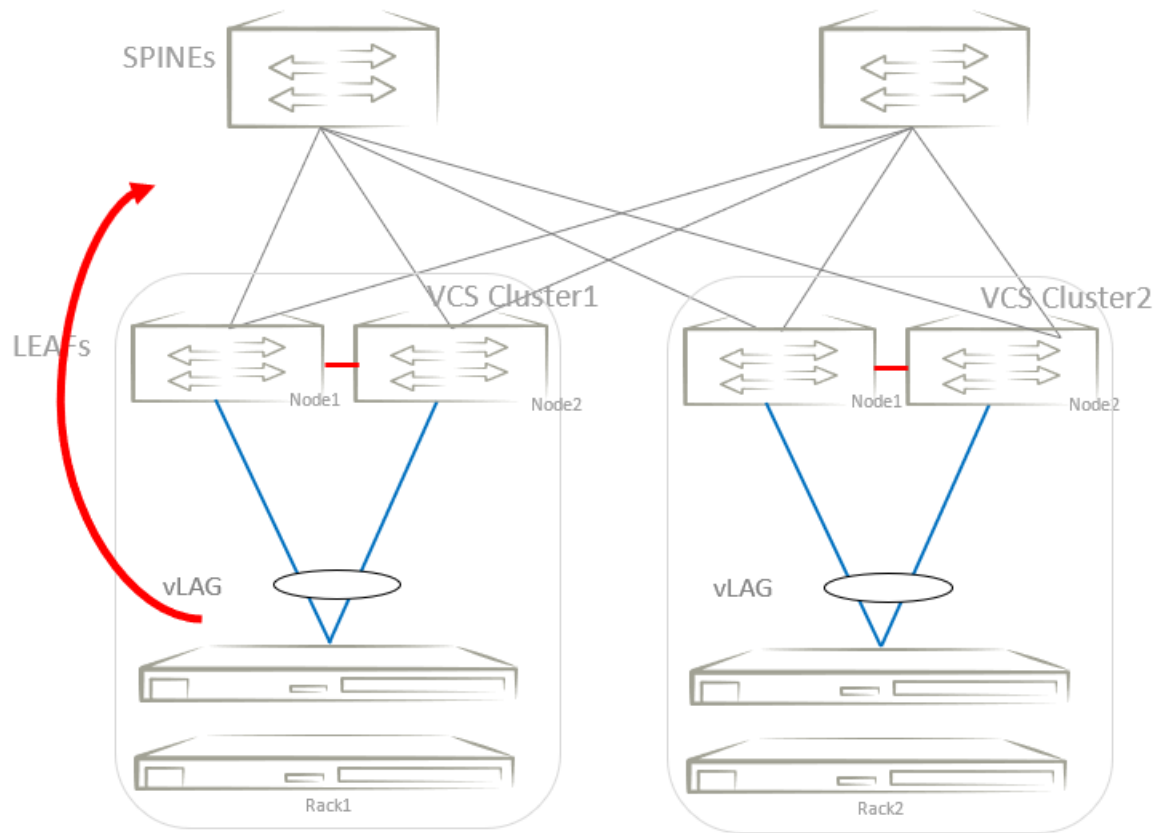


Figure 1: Scenario 1

Scenario 2

Node1 in VCS Cluster1 is reloading. Routing protocols between leaf and spine nodes could be converging before all tunnels are formed in Node1. Spine nodes could start load balancing the overlay traffic to Node1, but all this traffic could be dropped as the tunnels are not yet formed in Node1.

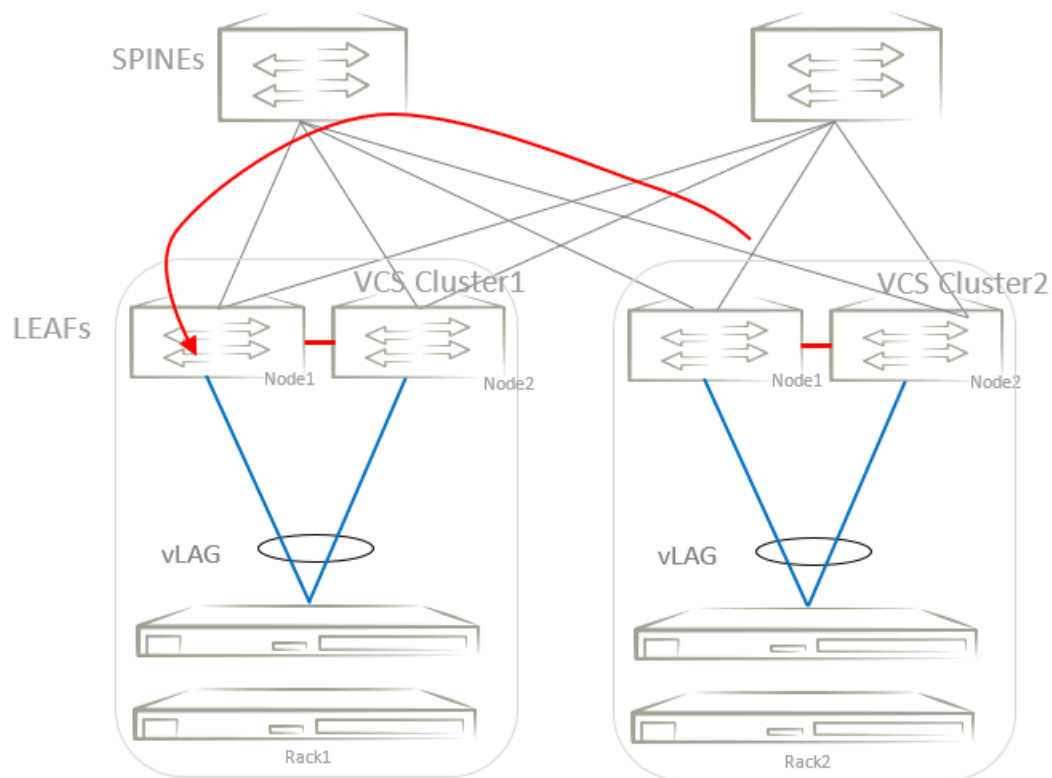


Figure 2: Scenario 2

After a switch reload, interfaces on which reload-delay is enabled remain administratively down for at least the delay-time configured by the user. After this time, the interface becomes administratively up.

For graceful vLAG host-traffic restoration, the reload delay must be configured on client interfaces such as physical interfaces and vLAG/port-channel interfaces. For graceful spine-traffic restoration, reload delay is configured on a loopback interface whose IP address is used as the source IP address of a tunnel end point. The routing protocols become aware of the tunnel interfaces only after the specified reload-delay time, after the tunnel has been established. This avoids traffic black-holing.

Configuration examples

This example enables reload delay and specifies an optional delay time of 1200 seconds on a port-channel.

```
device# configure terminal
device(config)# interface port-channel 10
device(config-Port-channel)# reload-delay enable 1200
```

This example enables reload delay and specifies a delay time on an Ethernet interface.

```
device# configure terminal
device(config)# int eth 0/12
device(conf-if-eth-0/12)# reload-delay enable 1600
Newly configured reload delay value will be applicable after system reload.
device(conf-if-eth-0/12)#
```

This example enables reload delay and specifies a delay time on a loopback interface.

```
device# configure terminal
device(config)# interface loopback 10
device(config-lo-10)# reload-delay enable 1200
```

This example specifies a global reload-delay time of 1800 seconds. (The interface configuration always takes precedence over the global configuration.)

```
device# configure terminal
device(config)# reload-delay 1800
```

This example uses the **show interface port-channel** command to display the configuration on a port-channel.

```
device# show interface port-channel 10
Port-channel 10 is admin down, line protocol is down (admin down)
Hardware is AGGREGATE, address is d884.66e9.fb60
  Current address is d884.66e9.fb60
Interface index (ifindex) is 671088650 (0x2800000a)
Minimum number of links to bring Port-channel up is 1
MTU 1548 bytes
LineSpeed Actual      : Nil
Allowed Member Speed : 10000 Mbit
Priority Tag disable
Forward LACP PDU: Disable
Route Only: Disabled
Tag-type: 0x8100
Reload delay time: 1200, Remaining time: 975
Last clearing of show interface counters: 00:03:45
Queueing strategy: fifo
Receive Statistics:
```

Chassis and host names

A device can be identified by its IP address or by its host name and chassis name. You can customize the host name and chassis name. You can also change the default chassis IPv4 or IPv6 address.

Customizing chassis and host names

Extreme recommends that you customize the chassis name for each device. Some system logs identify the device by its chassis name; if you assign a meaningful chassis name, logs are more useful. You can also configure the host name.

To customize the chassis name and host name, perform the following steps:

1. In privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Configure the chassis name.

```
device(config)# switch-attributes chassis-name SLX-market1
```

A chassis name can be from 1 through 30 characters long, must begin with a letter, and can contain letters, numbers, and underscore characters.

The default chassis name is SLX9540-# where # is the number of slots in the chassis.

3. Configure the host name.

```
device(config)# switch-attributes host-name SLX-mrkt
SLX-mrkt(config)#
```

This example changes the host name to SLX-mrkt and it is displayed in the prompt.

A host name can be from 1 through 30 characters long. It must begin with a letter, and can contain letters, numbers, and underscore characters. The default host name is SLX.

4. Exit global configuration mode.

```
SLX-mrkt(config)# exit
```

5. Verify the configuration.

```
SLX-mrkt# show running-config switch-attributes
switch-attributes chassis-name SLX-market1
switch-attributes host-name SLX-mrkt
!
```

The following configuration is an example of the previous steps.

```
device# configure terminal
device(config)# switch-attributes chassis-name SLX-market1
device(config)# switch-attributes host-name SLX-mrkt
SLX-mrkt(config)#
```

System clock

The operation of the device does not depend on the date and time and the Extreme device with an incorrect date and time value functions properly. However, since logging, error detection, and troubleshooting use the date and time, you should set the clock correctly.



Note

You can set the system clock if there are no NTP servers configured. Otherwise, an active NTP server, if configured, automatically updates and overrides the system clock.

Setting the clock

The Extreme device allows you to manually set the system clock. The time counter setting is retained across power cycles.

To set the clock, perform the following step:

1. In privileged EXEC mode, set the current date and time in the UTC timezone.



Note

This **must** be set to the UTC time, otherwise configuration of the timezone will cause the system to adopt the incorrect local time.

```
device# clock set 2019-12-10T16:38:00
```

This example sets the time and date to 16:38:00 on December 10, 2019.



Note

Setting the clock is not required when NTP is configured and the clock is synchronized to an external NTP server.

2. Enter global configuration mode.

```
device# configure terminal
```

3. Change the time zone.

```
device(config)# clock timezone America/Los_A
device(config)# end
device# show clock
2019-12-10 08:38:18 America/Los_Angeles
device#
```

This example changes the time zone to the region of America and the city of Los Angeles.

The following configuration is an example of the previous steps.

```
device# clock set 2019-12-10T16:38:00
device# conf
Entering configuration mode terminal
device(config)# clock timezone America/Los_A
device(config)# end
device# show clock
2019-12-10 08:38:18 America/Los_Angeles
device#
```

Management VRFs

Virtual Routing and Forwarding (VRF) is a technology that controls information flow within a network, isolating the traffic by partitioning the network into different logical VRF domains.

All management services on the Extreme device are VRF aware. The management services can select a particular VRF to reach a remote server based on a VRF. The VRFs are management (mgmt-vrf), default (default-vrf), and user defined VRF (user-vrf).

By default, the device creates a VRF for management named mgmt-vrf and, all manageability services are accessible through this VRF. Multiple instances of IP services can be instantiated in multiple VRFs. For example, SSH can be in more than one VRF. IP services can have up to five VRF instances.

VRF reachability

The Extreme device supports the VRF reachability service. Reachability determines which VRF contains the routing information needed to reach the application servers. For example, when you configure an SSH server, you can configure the VRF information for the VRF context to resolve the SSH server route.

VRF reachability indicates the details of the VRF for servicing requests from the clients. It also indicates the clients specifying the VRF for reaching a source to ensure that the management packets are serviced or routed in a server VRF domain.

These two types of reachability services are also referred to as device-initiated and server-based services.

VRF reachability for device-initiated services

The following table lists the device-initiated services and associated commands that VRF reachability supports.

Table 5: Device-initiated services and associated commands that VRF reachability supports

Service	VRF-related command	Additional information
Firmware download	firmware download [default-config] ftp scp sftp [use-vrf <i>vrf-name</i>]...	Uses TCP IPv4 or IPv6. By default, mgmt-vrf is used: use-vrf is optional.
LDAP	ldap-server host { <i>ip-address</i> <i>host_name</i> } [use-vrf <i>vrf-name</i>] [port <i>portnum</i>] [ldaps] [timeout <i>seconds</i>] [retries <i>num</i>] [basedn <i>base domain name</i>]	Default vrf is mgmt-vrf . use-vrf is optional.
Logging server	logging syslog-server { ipv4 ipv6 <i>address</i> } [use-vrf <i>vrf-name</i>]	Uses TCP UDP IPv4 or IPv6.
NTP	ntp server <i>ip-address</i> [use-vrf <i>vrf-name</i>]	Uses NTP UDP IPv4.
RADIUS	radius-server host <i>host-name</i> [use-vrf <i>vrf-name</i>]	Uses UDP IPv4 or IPv6
sFlow	[no] sflow collector <i>ipv4/ipv6 address port-number</i> [use-vrf <i>vrf-name</i>] [no] sflow source-interface <i>interface-type interface-number</i>	Uses sFlow UDP IPv4 or IPv6. In the case of the sflow source-interface command, the VRF of the specified interface is used. Note: Any given interface can belong to only one VRF at any given time.
SSH Client	ssh { <i>IP_address</i> <i>hostname</i> } [-l] <i>remote user/login name</i> [vrf <i>vrf-name</i>]	VRF is optional; default-vrf is used by default.
SNMP notification	snmp-server host <i>ip-address</i> [use-vrf <i>vrf-name</i>] snmp-server v3host <i>ip-address</i> [use-vrf <i>vrf-name</i>]	Uses SNMP UDP IPv4 or IPv6. By default, mgmt-vrf is used to send the SNMP notifications.
Support save	copy support { ftp scp } [use-vrf <i>vrf-name</i>] ...	Uses TCP IPv4 or IPv6. By default, mgmt-vrf is used and a user-defined VRF is optional.
TACACS+	tacacs-server host <i>host-name</i> [use-vrf <i>vrf-name</i>]	Uses TCP IPv4 or IPv6
Telnet client	telnet <i>IP_address</i> <i>hostname</i> [vrf <i>vrf-name</i>]	VRF is optional; default-vrf is used by default.

All these implementations use forward referencing of the VRF name in the **use-vrf** option, unless noted. At runtime when making the socket connection, the VRF ID by name must be resolved. If it does not resolve, it will result in a connection error.

VRF reachability for server-based services

The server services running on the Extreme device must listen to the requests in all the VRFs or a specified VRF and send the response back to the client in the same VRF where the request arrived. Thus, the services can come through any in-band interface bound to any VRF.

Each server-based service can have a maximum of 32 VRF instances; one mgmt-vrf, one default-vrf, and 30 user-defined VRFs. The following table lists the server services and associated commands that VRF reachability supports.



Note

The SNMP server listens on all VRFs and sends the response back on the same VRF where the request arrived.

HTTP and HTTPS are mutually exclusive on the Extreme device and both will not be enabled in different VRFs.

Table 6: Server-based services and associated commands that VRF reachability supports

Service	VRF-related command	Additional information
HTTP	<code>[no] http server [use-vrf vrf-name] [shutdown]</code>	By default, the HTTP service is associated and started on mgmt-vrf and default-vrf. A user-defined VRF is optional.
SSH	<code>[no] ssh server [use-vrf vrf-name] shutdown</code>	By default, the SSH service is associated and started on mgmt-vrf and default-vrf. A user-defined VRF is optional.
Telnet	<code>[no] telnet server [use-vrf vrf-name] shutdown</code>	By default, the Telnet service is associated and started on mgmt-vrf and default-vrf. A user-defined VRF is optional.

Telnet, HTTP, and SSH limitations and considerations

Telnet, HTTP, and SSH limitations and considerations are as follows:

- By default, SSH and Telnet services are associated and started on mgmt-vrf and default-vrf.
- You cannot remove mgmt-vrf from the SSH and Telnet services.
- Telnet and SSH server can be enabled on a maximum number of 32 VRFs.
- SSH and Telnet Services started on VRF context is applicable for both IPv4 and IPv6 addresses.
- A maximum of 32 user logins are allowed in the device. These sessions are a cumulative count of login sessions through SSH and Telnet across all the configured VRFs.
- Inter-VRF route leaking is not supported for SSH, HTTP, HTTPS, and Telnet. When you try to use any of these services to access a leaked VRF (user-vrf), the connection is refused. In addition, the access

service ceases to function correctly on the local VRF (mgmt-vrf) and you must restart it; for example, to restart the SSH service on the local VRF, run the **ssh server restart** command.

Zero Touch Provisioning

Zero Touch Provisioning (ZTP) is an automated process that uses the DHCP process to download firmware and set up the device configuration.



Note

The Zero Touch Provisioning feature is supported on the following platforms:

- SLX9540
- SLX9640
- SLX-9250
- SLX-9150-48Y
- SLX-9150-48XT

Zero Touch Provisioning (ZTP) is an automated process that uses the DHCP process to download firmware and set up the device configuration.

The ZTP process eliminates the need to log in manually to the console to bring up the device with the correct firmware and required configuration. When the device is in the factory default configuration, ZTP can start automatically upon device bootup.

This process reduces the time taken for firmware download and device configuration. All switches download the same firmware and configuration script from the ZTP configuration file.

The following configuration considerations apply to ZTP:

- ZTP is not supported for customers who do not use DHCP.
- ZTP supports only DHCPv4.
- The DHCP server must be configured with GET options 66 and 67 to set the ZTP configuration file.
- ZTP is triggered on a new device by means of Open Network Install Environment (ONIE), by means of the **write erase** command.
- ZTP supports both in-band ports and management interfaces in the management VRF.
- After ZTP completes, all the in-band ports return to the default VRF state.
- To establish network connectivity, ZTP retries indefinitely to establish a network connection among in-band ports and management interfaces until the firmware download completes, downloading all firmware packages before the device reboots.
- The interface is selected when it passes the sanity test. The order of selection is based on the response order of GET options 66 and 67 during the DHCP server detection process.
- All the interfaces are scanned in parallel to detect DHCP options 66 and 67.
- If ZTP is enabled and there is no DHCP server configured with options 66 and 67 for ZTP, the device indefinitely tries to discover a DHCP server. The user must disable ZTP by using the **dhcp ztp cancel** command and must reboot the device before applying any configuration.
- Network connectivity through the management interface has higher priority over connectivity through in-band ports.

- ZTP is supported only in standalone mode.
- Customer configurations are supported with the Python script.
- The ZTP configuration file supports both a common setting or device-specific settings.
- The ZTP progress is displayed on the serial console and is saved in a log file.
- The DHCP client ID of the device must be set up in the device-specific ZTP configuration file.
- The RASlog is disabled during the early stages of the ZTP process.
- Breakout ports are not supported, because a device reboot is required.
- Only the default speeds (10 or 100 G) on in-band ports are supported for the ZTP process.

Routing for ZTP

ZTP supports FTP and HTTP to fetch ZTP configurations, scripts, switch startup configurations, and firmware.

The DHCP and FTP/HTTP server may not be reachable by all the nodes in the IP Fabric. A route must be configured on the first-level node with a connection to DHCP and FTP/HTTP servers. ZTP must first be run on the first-level node by means of the Python script to enable iphelp to forward the traffic to the servers. The ZTP process can then run on the next-level nodes. Eventually the farthest nodes can connect to the servers for ZTP.

Using ZTP

Follow these steps to enable ZTP in standalone mode.

1. Establish a network connection with cable on one-to-multiple in-band ports or a management interface.
2. Power up the device in the factory default configuration or run the **write erase** command from the SLX-OS CLI.
3. On device boot up, the ZTP process performs the following actions:
 - a. Disables the RASlog to the serial output.
 - b. Enables in-band ports in the management VRF.
 - c. Detects a DHCP server with option 66 or 67 configured over a management interface and all in-band port interfaces.
 - d. Selects one interface not used before to assign the DHCP IP address.
 - e. Downloads the ZTP configuration file from the FTP/HTTP server.
 - f. Validates the ZTP configuration file.
 - g. If required, ZTP performs a firmware download. Firmware download reboots the device automatically. If no firmware download is needed, the ZTP process continues to configure the switch.

In the current state the ZTP process returns to substep (b) in the following situations:

- If there is a failure in any of the above-mentioned substeps from (b) through (g)
- If the device reboots from the CLI
- If the device crashes

On device bootup, the continuation of the ZTP process is indicated on the console. Wait for firmware commit to complete. If the firmware commit fails, the ZTP process aborts. If the script is enabled, the script is launched automatically.

4. Enable the RASLog.

For more information and log outputs for canceling DHCP ZTP, refer to the *SLX-OS Command Reference Guide*.

ZTP configuration

To manage devices, the DHCP server and the FTP server must be set up to provide the environment.

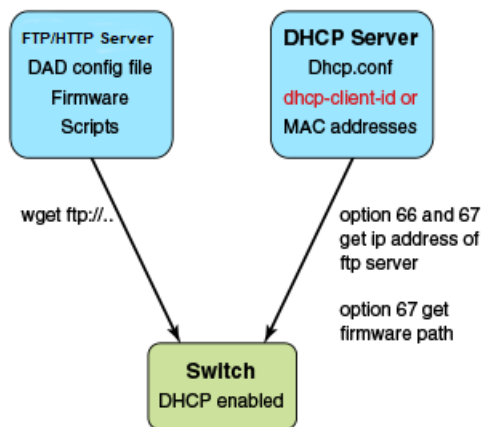


Figure 3: ZTP configuration

DHCP server

DHCP Server version 4.2.4 was tested on Ubuntu 14.04 (Trusty). The `dhcpd.conf` file must have option 66 (TFTP Server Name) and option 67 (Filename) set for ZTP. Option 66 is used for the FTP server IP address or host name. Option 67 is used for the ZTP configuration file path.

When the device starts the DHCP process, it sends the DHCP client ID to the DHCP server to get the IP address and options 66/67. The device then downloads the ZTP configuration file from the FTP server. To set up a different ZTP configuration file for different devices, the DHCP Client ID can be used in the `dhcpd.conf` file. Whenever `dhcpd.conf` is changed, the `dhcpd` server must be restarted.

FTP server

vsFTP server version 3.0.2 was installed and tested on Ubuntu 14.04 (Trusty). The FTP server stores the ZTP configuration file, firmware, switch configuration file, or Python script. The location of these configuration files under the FTP server base directory is flexible.

HTTP server

Apache server version 2.4.18 was installed and tested on Ubuntu 14.04 (Trusty). The HTTP server stores the ZTP configuration file, firmware, switch configuration file, or Python script. The location of these configuration files under the HTTP server base directory is flexible.

ZTP configuration script

The ZTP process can run the script to set up the device configuration automatically. For now, only the Python script is supported. The script takes no parameters.

The script can automate any command line, including SLX-OS and Linux commands, such as the configuration download command, **copy ftp:// . . . running-config**.

ZTP configuration file

The ZTP configuration file has two configuration sections: common and device-specific. The common section is shared by all the switches in the IP Fabric. The settings in the device-specific section can be used for a single switch or a group of switches with the DHCP client ID. If the `host_client_id` string matches the starting substring of the DHCP client ID of the switch, the device-specific section is used by the switch.

Python script example

The following is an example Python script.

```
# !usr/local/python/3.3.2/bin/python3
import os
import sys, getopt

def main(argv):
    log.write("apply config\n")
    # change login banner
    CLI("conf ; banner login DAD ; end")
    # config download
    CLI("copy scp://root:extr123@192.169.0.2/castorT.startup.cfg running-config")
if __name__ == "__main__":
    main(sys.argv[1:])
```

FTP server configuration file

The following is an example FTP server configuration file.

```
local_enable=YES
write_enable=YES
local_umask=022
dirmessage_enable=YES
xferlog_enable=YES
connect_from_port_20=YES
xferlog_std_format=YES
listen=NO
listen_ipv6=YES

pam_service_name=vsftpd
userlist_enable=NO
tcp_wrappers=YES

# dad settings
anonymous_enable=YES
no_anon_password=YES
anon_root=/var/ftp
delay_failed_login=30
max_clients=100
anon_max_rate=8388608
```


DHCP server configuration file

The following is an example DHCP server configuration file, dhcp.conf

```
# ddns-update-style standard;
ddns-update-style interim;
ddns-ttl 600;
ignore client-updates; # Overwrite client configured FQHNs
ddns-domainname "infralab.com.";
ddns-rev-domainname "in-addr.arpa.";

option ntp-servers 192.168.0.2;
option domain-name-servers 192.168.0.2;
option domain-name "infralab.com";
option domain-search "infralab.com";

default-lease-time 600;
max-lease-time 7200;

authoritative;

log-facility local7;

key "extr-key" {
    algorithm hmac-md5;
    secret
    "dtBgNTAoaqZmwV5c4SueybjOvhe6OIqgacluQrzGBv5O4X4nIEBEEGWRf0lCnbFhuIJXGExNBjDdNSqgBMeNI8w=="
;
};

subnet 192.168.0.0 netmask 255.255.255.0 {
    range 192.168.0.100 192.168.0.200;
    option subnet-mask 255.255.255.0;
    option broadcast-address 192.168.0.255;
    zone 0.168.192.in-addr.arpa. {
        primary 192.168.0.2;
        key "extr-key";
    }
    zone infralab.com. {
        primary 192.168.0.2;
        key "extr-key";
    }
}
# cluster switches
group{
    option bootfile-name "/config/unified-cfg.min";
    option tftp-server-name "192.168.0.2";
    option routers 192.168.0.2;

    # sw0
    host sw0 {
        option dhcp-client-identifier = "EXTREMENETWORKS##SLX9240##EXG3342L00V";
        hardware ethernet 52:54:00:0E:95:8B;
        fixed-address 192.168.0.90;
    }
# fixed ip address
```

ZTP configuration file

The following example has three sections: common, switch 1, and switch 2.

```
version=3
date=03/20/2018
supported_nos=17s.1.03

common_begin
```

```

vcsmode=SA
scriptcfgflag=2 #0-config file only, 1-script only, 2 both
script=/script/ztp.py
fwdir=/fw/slxos17s.1.03_bld04
common_end

# model SLXL9140 hosts
host_client_id=EXTREMENETWORKS##SLX9140
script=/script/Frreedomlic.py
startup=/config/freedomlic.cfg
host_end

# model SLX9140 with serial number
host_client_id=EXTREMENETWORKS##SLX9140##EXH3327M014
startup=/config/freedom_ospf.cfg
script=/script/FreedomZTP.py
host_end

# model Accton hosts with serial number
host_client_id=EXTREMENETWORKS##ModelNumber##SerialNumber
startup=/config/AcctonConfig.cfg
script=/script/AcctonZTP.py
host_end

```

ZTP configuration file definitions

The following table contains the ZTP configuration file definitions.

Table 7: ZTP configuration file definitions

Variable description	Description
version	Only version 3 is supported.
date	The last modified date.
supported_nos	The release firmware version supporting the ZTP configuration file.
host_client_id, host_end	Host_client_id marks the beginning of the section host_end marks the end. User could set up the switch specific section with full dhcp client id or its prefix. Ex. <i>host_client_id=EXTREMENETWORKS##SLX9140#EXH3319M01J</i> <i>script=/script/dad1new.py</i> <i>host_end</i>
common_begin, common_end	The setting in the section will be shared by all switches.
vcsmode=SA	Only standalone mode is supported.

Table 7: ZTP configuration file definitions (continued)

Variable description	Description
vcstimeout	If omitted, the default is 60 minutes. The timeout to wait for ZTP to complete configuration file download or Python script. If the configuration download process or Python script has issues, the zero touch provisioning process will stop the download after timeout and claim that ZTP is complete. You will need to increase the timeout if configuration download or Python script takes a long time to complete.
fwdir	Firmware path in the FTP/HTTP server. For example Fwdir=/fw/slxoss17r.1.00_bld34. If base directory of the server is /var/ftp, then the absolute path of firmware in ftp server is located at /var/ftp/fw/slxoss174.1.00_bld34.
startup	The path to the switch configuration file in the FTP server. If omitted, the switch will take the default configuration. The value can be "default" or user configuration file.
scriptcfgflag	The default is 0, when not specified. The meaning of the value is: 0 - only use startup, script is ignored 1 - only use script, startup is ignored
script	The device configuration Python script file.

ZTP commands

ZTP has two commands, **dhcp ztp log** and **dhcp ztp cancel**. These are illustrated below.

The following displays current ZTP progress.

```
device# dhcp ztp log
ZTP, Sat Feb 17 02:48:51 2001, ===== ZTP start =====
ZTP, Sat Feb 17 02:48:51 2001, disable raslog
ZTP, Sat Feb 17 02:48:51 2001, CLI is ready
ZTP, Sat Feb 17 02:49:19 2001, inband ports are enabled
ZTP, Sat Feb 17 02:49:19 2001, serial number = 771232X1750017
ZTP, Sat Feb 17 02:49:19 2001, model name = AS7712-32X
ZTP, Sat Feb 17 02:49:19 2001, use both management interface and inband interfaces
ZTP, Sat Feb 17 02:49:19 2001, checking inband interfaces link status
ZTP, Sat Feb 17 02:49:19 2001, find link up on interfaces: eth0
ZTP, Sat Feb 17 02:49:19 2001, start dhcp process on interfaces: eth0
ZTP, Sat Feb 17 02:49:20 2001, interface eth0 receives dhcp response
ZTP, Sat Feb 17 02:49:20 2001, ping server 192.169.0.1
ZTP, Sat Feb 17 02:49:21 2001, ping succeed
ZTP, Sat Feb 17 02:49:21 2001, download ZTP config file from https://192.169.0.1/config/ztp.conf
ZTP, Sat Feb 17 02:49:21 2001, download ZTP config file from http://192.169.0.1/config/ztp.conf
ZTP, Sat Feb 17 02:49:21 2001, receive ZTP configuration file [ztp.conf]
ZTP, Sat Feb 17 02:49:21 2001, interface eth0 connectivity test pass
ZTP, Sat Feb 17 02:49:21 2001, download switch config file [startup.cfg]
ZTP, Sat Feb 17 02:49:21 2001, ZTP configuration sanity check pass
ZTP, Sat Feb 17 02:49:22 2001, skip firmware upgrade
```

```

ZTP, Sat Feb 17 02:49:38 2001, replay config file...
ZTP, Sat Feb 17 02:50:25 2001, commit configuration
ZTP, Sat Feb 17 02:50:25 2001, ZTP succeed
ZTP, Sat Feb 17 02:50:25 2001, enable raslog
ZTP, Sat Feb 17 02:50:25 2001, ===== ZTP completed =====

device# dhcp ztp cancel
Warning: This command will terminate the existing ZTP session
Do you want to continue? [y/n] y

```

The following cancels the current ZTP session.



Note

Before making any configuration changes from the CLI, the user must reboot the switch to return to the default configuration. A reboot abandons all switch configuration set by ZTP.

```

device# dhcp ztp cancel
Warning: This command will terminate the existing ZTP session
Do you want to continue? [y/n] y

```

Example of ZTP in a two-node topology

This section describes ZTP IP routing in a two-node topology.

In the following figure, Switch 1 Eth 0/8 has direct connection to the DHCP or FTP/HTTP server. Switch 1 acts as a router for Switch 2 to reach the DHCP or FTP/HTTP server. A default route on Switch 1 is configured on the server for traffic sent from the DHCP server to reach Switch 2 (see the default route below). External access to the DHCP server is on Eth 0. There are two configurations for Switch 1:

- One is set up on Eth 0/8 by the DHCP server for ZTP to establish a connection to the DHCP server to download the ZTP configuration file.
- The other is set up by the Python script to configure Switch 1 as a router with Eth 0/8 to the server and Eth 0/3 to Switch 2.

DHCP relay is configured on Eth 0/3 in Switch 1 for DHCP requests from Switch 2. Switch 1 Eth 0/8 and Eth 0/3 must be in different subnets.

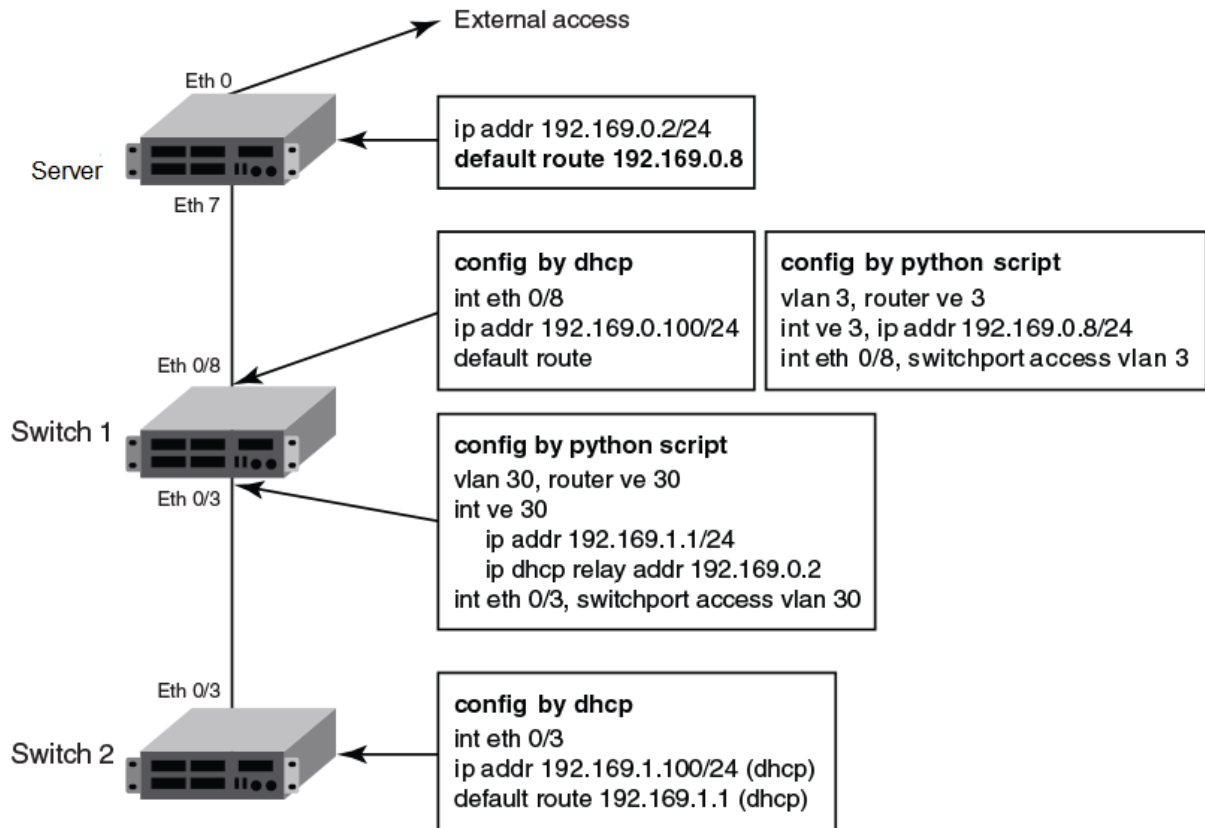


Figure 4: ZTP two-node topology



Note

The Python script for Switch 1 should be manually tested to verify the routing configuration before ZTP is started for Switch 2.

The DHCP server configuration has two subnet address pools, based on the DHCP client ID: "level_1" for Switch 1 and "level_2" for Switch 2, as in the following example.

```
class "level_1" {
    match if option dhcp-client-identifier = "EXTREMENETWORKS##SLX9140##EXH3319M01J";
    <EXH3319M01J is the device serial number>
}
class "level_2" {
    match if option dhcp-client-identifier = "EXTREMENETWORKS##SLX9140##EXH3314M00L";
    <EXH3314M00L is the device serial number>
}
subnet 192.169.0.0 netmask 255.255.255.0 {
    pool {
        allow members of "level_1";
        range 192.169.0.100 192.169.0.200;
    }
    option bootfile-name "/config/ztp.cfg";
    option tftp-server-name "192.169.0.2";
    option routers 192.169.0.2;
    option subnet-mask 255.255.255.0;
    option broadcast-address 192.169.0.255;
}
subnet 192.169.1.0 netmask 255.255.255.0 {
    pool {
        allow members of "level_2";
```

```

        range 192.169.1.100 192.169.1.200;
    }
    option bootfile-name "/config/ztp.cfg";
    option tftp-server-name "192.169.0.2";
    option routers 192.169.1.1; ip address as routers in level 2 subnet
    option subnet-mask 255.255.255.0;
    option broadcast-address 192.169.1.255;
}

```

Configuration file example

The following is an example configuration file.

```

version=3
date=04/29/2016
supported_nos=17s.1.00 17r.1.00

common_begin
vcsmode=SA
fwdir=/bld/Nightly_nos_fusion_davinci_dev_160822_0600/dist
scriptcfgflag=2 #0-config file only, 1-script only, 2 both
script=/script/ztp.py
common_end

# model SXL9140 hosts
host_client_id=EXTREMENETWORKS##SLX9140 □ for Switch 2
startup=/config/startup.cfg
host_end

# switch 1 as router node
host_client_id=EXTREMENETWORKS##SLX9140##EXH3319M01J
startup=/config/freedom1_ospf.cfg
host_end

```

Configuration flow

The following sequence summarizes the configuration flow:

1. Execute the **write erase** command from the CLI on both Switch 1 and Switch 2 simultaneously.
2. Switch 1 behaves as a single-node ZTP switch.
3. Switch 2 is delayed in detecting the DHCP server by means option 66 or 67 until ZTP on Switch 1 succeeds, so that the static route is configured successfully. If Switch 1 fails, Switch 2 waits indefinitely.
4. If ZTP is enabled, it shows the ZTP progress log as follows:

```

device# dhcp ztp log
ZTP, Wed Jun 29 17:32:36 2016, ===== ZTP start =====
ZTP, Wed Jun 29 17:32:36 2016, disable raslog
ZTP, Wed Jun 29 17:32:36 2016, CLI is ready
ZTP, Wed Jun 29 17:33:11 2016, inband ports are enabled
ZTP, Wed Jun 29 17:33:11 2016, serial number = EXH3343L014
ZTP, Wed Jun 29 17:33:11 2016, model name = SLX9140
ZTP, Wed Jun 29 17:33:11 2016, use inband interfaces only
ZTP, Wed Jun 29 17:33:13 2016, get link down on all the interfaces
ZTP, Wed Jun 29 17:33:13 2016, retry in 10 seconds
ZTP, Wed Jun 29 17:33:23 2016, inband ports are enabled
ZTP, Wed Jun 29 17:33:24 2016, serial number = EXH3343L014
ZTP, Wed Jun 29 17:33:24 2016, model name = SLX9140
ZTP, Wed Jun 29 17:33:24 2016, use inband interfaces only
ZTP, Wed Jun 29 17:33:24 2016, get link down on all the interfaces
ZTP, Wed Jun 29 17:33:24 2016, retry in 10 seconds

```

```

ZTP, Wed Jun 29 17:33:34 2016, inband ports are enabled
ZTP, Wed Jun 29 17:33:34 2016, serial number = EXH3343L014
ZTP, Wed Jun 29 17:33:34 2016, model name = SLX9140
ZTP, Wed Jun 29 17:33:34 2016, use inband interfaces only
ZTP, Wed Jun 29 17:33:35 2016, checking inband interfaces link status
ZTP, Wed Jun 29 17:34:25 2016, find link up on interfaces: Eth0.6 Eth0.8
ZTP, Wed Jun 29 17:34:25 2016, start dhcp process on interfaces: Eth0.6 Eth0.8
ZTP, Wed Jun 29 17:34:34 2016, interface Eth0.8 receives dhcp response
ZTP, Wed Jun 29 17:34:34 2016, config ip address 192.169.0.147/24 on interface Eth0.8
ZTP, Wed Jun 29 17:34:39 2016, ping ftp server 192.169.0.2
ZTP, Wed Jun 29 17:34:40 2016, ping succeed
ZTP, Wed Jun 29 17:34:41 2016, download ZTP config file from ftp://192.169.0.2/config/
ztp.cfg
ZTP, Wed Jun 29 17:34:41 2016, receive ZTP configuration file [ztp.cfg]
ZTP, Wed Jun 29 17:34:41 2016, interface Eth0.8 connectivity test pass
ZTP, Wed Jun 29 17:34:41 2016, download script file [ztp.py]
ZTP, Wed Jun 29 17:34:41 2016, ZTP configuration sanity check pass
ZTP, Wed Jun 29 17:38:22 2016, ===== ZTP continue =====
ZTP, Wed Jun 29 17:38:22 2016, disable raslog
ZTP, Wed Jun 29 17:38:22 2016, CLI is ready
ZTP, Wed Jun 29 17:38:58 2016, running configuration script [ztp.py]
ZTP, Wed Jun 29 17:39:25 2016, commit configuration
ZTP, Wed Jun 29 17:39:25 2016, ZTP succeed
ZTP, Wed Jun 29 17:39:25 2016, enable raslog
ZTP, Wed Jun 29 17:39:25 2016, ===== ZTP completed =====
device# dhcp ztp cancel
device# dhcp ztp cancel

Warning: This command will terminate the existing ZTP session
After ZTP has been confirmed canceled, you need to run "reload system" before
configuring the switch.
Do you want to continue? [y/n]

```



Note

ZTP is enabled by default for switch in factory default or after running "write erase". User must cancel ZTP and reload system. After switch restarts, switch is ready for all commands.

ZTP session is designed to retry forever to detect the DHCP server and establish network connection for firmware download. If it is in the middle of firmware download, firmware download is completed successfully and the switch is in normal mode.

Limitation

1. If firmware download has not started yet, user should reboot the switch manually for normal mode.
2. If firmware download has already started, user should wait for firmware download to complete, before running any other commands, power cycle the switch, start a new firmware download, or to start a new ZTP session.
3. If firmware download completes and fails to reboot the switch, user should restart the switch manually for normal mode.

Enhanced Zero Touch Provisioning (ZTP+)

Enhanced Zero Touch Provisioning (ZTP+) is a method for devices to communicate with Extreme Management Center (XMC).

Enhanced Zero Touch Provisioning (ZTP+) allows an SLX device to verify the existing image, upgrade if necessary, or obtain the initial configuration from the Extreme Management Center (XMC). ZTP+ uses a

Cloud Connector plugin installed on the SLX device to communicate with XMC to set the following initial configuration.

- Image upgrade
- Static Management IP
- Gateway IP
- DNS Server IP and Domain Name
- HostName
- SNMP configuration (V1/V2/V3)
- NTP Server IP and Timezone Configuration

Pre-requisites and Dependencies

The following are the pre-requisites to install ZTP+.

- On-premise DHCP server configured to receive ZTP+ enabled devices.
- DNS configuration `extremecontrol.<domainname>`
- XMC/NetSight Server (Version 8.2 and above)
- SLXOS version 19.0.1 or higher

Please refer to the ZTP+ Device Configuration section in the XMC documentation for information to upload SLX firmware and configure ZTP+.

ZTP+ Phases of Operation

This section describes how the device determines whether to use ZTP or ZTP+ for configuration.

Connection

Once the SLX device is connected to the network, it uses DHCP to obtain an IP address. After receiving an IP address and DNS information via DHCP, the device tries to connect to XMC using the following means:

1. `extremecontrol.<customerDomainName>`
2. `devices.extremenetworks.com`
3. check DHCP options for original ZTP support

Discovery

When an SLX device comes up in ZTP mode, the native ZTP state machine uses the following procedure to determine which method to use for configuration; ZTP or ZTP+.

1. The CloudConnector plugin is launched in ZTP mode.
2. The DHCP options and network configuration are received from the DHCP Server.
3. If option 66 and 67 are received, proceed with native ZTP.
4. If the CloudConnector module discovers XMC in the network, proceed with ZTP+.
5. If option 66 and 67 are received but XMC is not discovered, then proceed with native ZTP state machine.

You can exit ZTP and ZTP+ mode at any point using the `dhcp ztp cancel` command.

The diagram below shows the ZTP to ZTP+ operational flow.

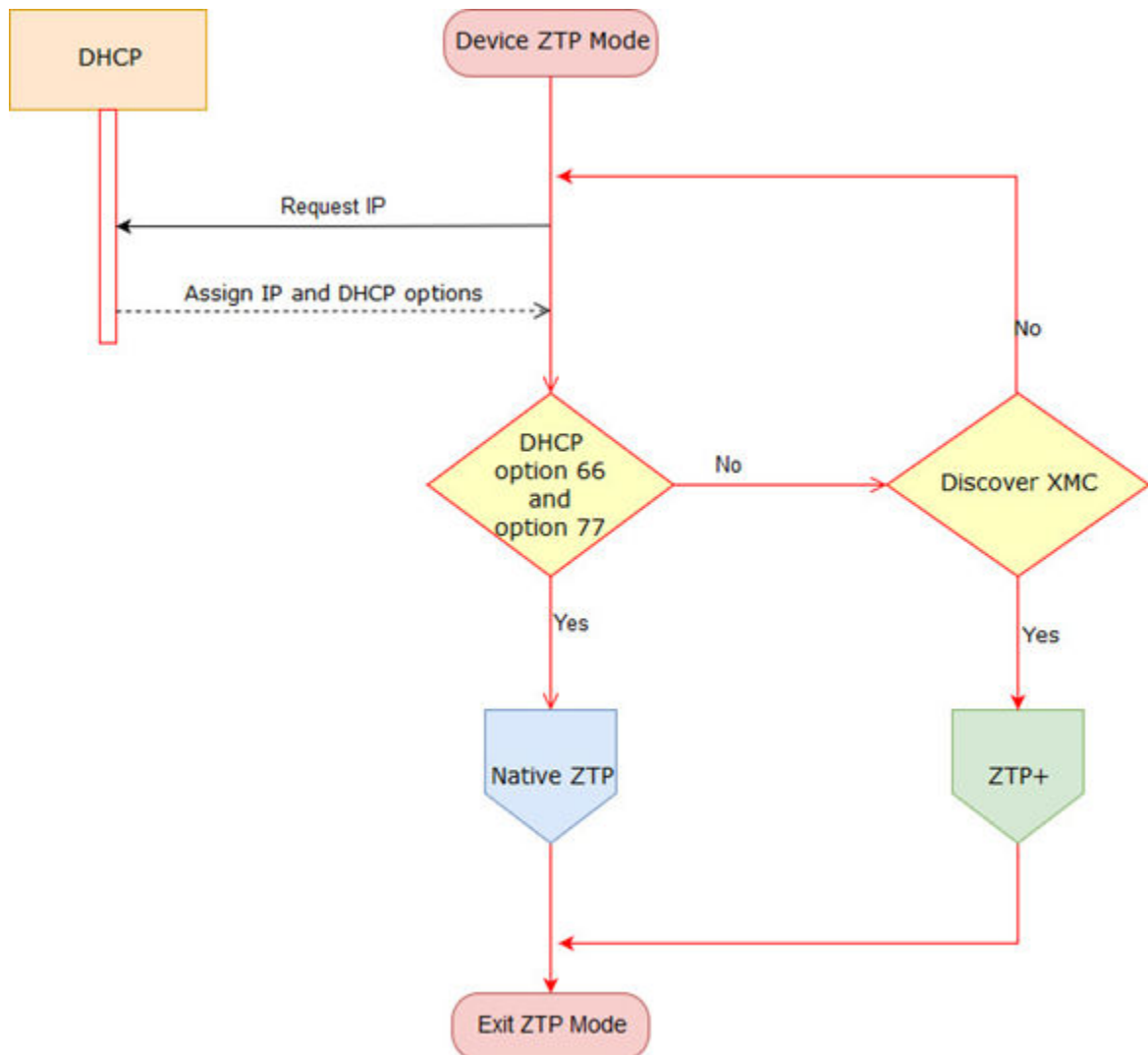


Figure 5: ZTP to ZTP+ Operational Flow

ZTP+ is functional only in the factory default or ZTP mode. On subsequent reboots with ZTP mode disabled, the CloudConnector will not be launched.

Firmware validation

The firmware validation process of ZTP+ verifies the device is running the appropriate version of software.

The CloudConnector and Extreme Management Center (XMC) exchange device software versions. If XMC detects a firmware update is required, it responds with the details to download the new version. The update or download of software is initiated by the device.

To identify the correct operating firmware in XMC, use Inventory Manager to assign the base operating system version as the Reference Image. The XMC ZTP+ server searches the reference image directory for all applications that need to be upgraded. These versions are used for the Application and OS

verification steps in ZTP+. If there are errors during this upgrade process, the device notifies XMC by posting an event to the server log., and retries the download operation.

Configuration

In the Configuration phase of ZTP+, the device notifies XMC that it is ready to receive its configuration.



Note

The device will not enter this stage until it has completed upgrading the software versions that were identified in the Firmware Validation process.

If the device is booting with factory defaults .i.e. `unconfigure switch all`, then no configuration block is sent. If the device has not yet been configured in XMC, the server issues a response to the device instructing it to periodically retry the configuration request until the user has applied a configuration to that device in XMC.

If the device has been previously configured (reboot), the device informs the server of its software version and current configuration when querying the server for configuration updates.

Please refer to the ZTP+ Device Configuration section in the XMC documentation for information to upload SLX firmware and configure ZTP+.

MAC address aging

MAC addresses that are dynamically learned are stored in MAC address table. The MAC address aging feature provides a mechanism to flush out the dynamic MAC addresses that remain inactive for a specified period.

The aging time of dynamic MAC address entries can be configured using the **mac-address-table aging-time** command. The MAC aging time can be configured to a value from 60 through 86400 seconds. By default, the aging time of dynamic MAC address entries is 300 seconds. The configured MAC aging time is applied to all MAC addresses in the system. You can disable the MAC address aging by specifying the aging time as 0 (zero).



Note

MAC address aging configuration per VLAN is not supported.

TCAM application-resource monitoring

Ternary Content Addressable Memory (TCAM) is specialized memory that stores complex tabular data and supports very rapid parallel lookups.

TCAM is used for storing different application filtering rules. These can be either L2, L3, or L4 control protocols. TCAM resources are used at different stages of the packet processor pipeline for providing the functionality. Some examples are:

- VT stage for inlif id
- TT stage for termination

- FWD stage for IPv6, IPv4 MC
- ACL - Ingress, Egress

A single lookup is performed per packet per TCAM bank.



Note

For devices based in the XGS chipset family, TCAM banks are referred to as *slices*. For a list of such devices, see "Supported Hardware".

It is possible to hit multiple TCAM banks for a single packet and the priority among the entries is selected based on either priority mode or interleaved mode. In priority mode bank1, entry1 takes precedence over bank2, entry2. In interleaved mode, minimal line entry is selected and first if both lines are equal. Associative data is 24b/48b when a TCAM bank is configured as 4K/2Kx80b/160b. When two TCAM banks are configured as 2K/128x320b model AD is 96b.

Table 8: (DNX devices) TCAM Ingress and Egress details

TCAM	12 TCAM banks and 4 Small banks. 3 key arrangement options per bank
TCAM (Shared by Ingress and Egress)	256 entries per small banks
	4K entries of 80 bits
	2K entries of 160 bits
	2K entries of 320 bits
	Concatenate two banks for a wider Key. The result options are:
	24 bits
	48 bits
	96 bits
	Accesses: 1 per packet per bank



Note

For devices based on the XGS chipset family, there are 12 TCAM slices, each containing 768 entries.

TCAM library-resource monitoring

The same TCAM hardware resources are shared by multiple applications, for example, PBR, IPv4 ACL. So, monitoring and reporting the threshold status for each of the applications at hardware is not possible. TCAM software library is created to abstract this single hardware resource shared by multiple applications.

To support this capability of TCAM, each resource has to go through the TCAM library code path of resource allocation to achieve monitoring.

Based on the allocation via TCAM library, resource usage statistics are collected and RAS logs are generated and associated with the specific TCAM application resources with flags as critical, warning

and info. Shared/fixed comment is present in the logs to reflect shared/fixed TCAM hardware resource by applications.

Hardware profiles

A variety of hardware profiles optimize ASIC resources for counters, port-channels, routes, and Ternary Content-Addressable Memory (TCAM)-allocation.



Note

When you change a hardware profile, the supported scale numbers remain the same with respect to the configuration even if hardware may not be able to fulfill them. This ensures that the same protocol and interface information remain valid with all hardware profile settings.

TCAM profiles

TCAM profiles enable you to optimize TCAM resources according to your system requirements.



Note

TCAM profiles other than default are supported only on devices based on the DNX chipset family. For a list of such devices, see "Supported Hardware".

TCAM is used by various forwarding applications. A TCAM profile supports a specified group of forwarding applications.

The following TCAM profiles are supported:

- default: Optimizes resources with basic support for all applications. MCT is supported.
- app-telemetry: Optimizes resources for application telemetry. MCT is supported.
- border-routing: Optimizes resources for border routing and BGP Flowspec features.
- cam-share: Enables TCAM sharing for security or policy-based routing (PBR) ACLs applied to multiple interfaces.
- layer2-ratelimit: Optimizes resources for Layer 2 ACL egress rate-limiting and related applications.
- (Not currently supported) multicast-profile: Optimizes resources for L2/3 IPv6 multicast.
- vxlan-visibility: Optimizes resources for VXLAN transit visibility and GRE.

TCAM Profile Scaling (DNX Devices)

For devices based on the DNX chipset-family, the following table displays maximum TCAM entries by features and TCAM profile.

	Features	default	border-routing	vxlan-visibility	app-telemetry	layer2-ratelimit	multicast-profile
Ingress	L2 ACL	4K*	2K	2K	6K*	6K*	2K*
	IPv4 ACL , rACLv4, PBRv4, BGP-FLOW-SPECv4		6K*	4K			
	IPv6 ACL, rACLv6, PBRv6, BGP-FLOW-SPECv6	2K		2K	NS	NS	4K
	VLL (PWE + VXLAN)	1.5K*	NS	NS	NS	NS	NS
	BUM-RL, Port-RL, VLAN-RL, BD-RL		1.5K**	1.5K**	1.5K**	1.5K**	2K**
	XC STAT	256	768	NS	NS	NS	NS
	Tunnel	4096	256	4096	256	256	256
	Application telemetry	NS	NS	NS	1K	NS	NS
Egress	L2 ACL	1K	1K	1K	1K	1K	1K
	IPv4 ACL	1K	1K	1K	1K	1K	1K
	L2 ACL-RL	NS	NS	NS	NS	2k	NS

Figure 6: TCAM-entries available per profile (DNX devices)

* For shared limits, the TCAM is filled using first-come first-served.

** DB is shared with L2 and L3 control protocol features.

NS = not supported.

RL = rate limiting.

BD = bridge domain.

Default TCAM Profile (XGS Devices)

For devices based on the XGS chipset-family, the following table displays maximum TCAM entries by features. The only TCAM profile supported is default.

	Features	TCAM Entries
Ingress	L2 ACL	501
	IPv4 ACL , rACLv4, PBRv4, BGP-FLOW-SPECv4	767
	IPv6 ACL, rACLv6, PBRv6, BGP-FLOW-SPECv6	767
	VLL (PWE + VXLAN)	767
	BUM-RL, Port-RL, VLAN-RL, BD-RL	
	XC STAT	NS
	Application telemetry	768
Egress	L2 ACL	256
	IPv4 ACL	256
	L2 ACL-RL	NS

Figure 7: Maximum TCAM entries by features

* For shared limits, the TCAM is filled using first-come first-served.

** DB is shared with L2 ctrl protocol, L3 protocol, and so forth.

NS = not supported.

RL = rate limiting.

BD = bridge domain.

Specifying a TCAM profile

Follow these steps to specify a TCAM profile.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **hardware** command to enable hardware configuration mode.

```
device(config)# hardware
```

3. Enter the **profile tcam** command to specify a TCAM profile.

```
device(config-hardware)# profile tcam multicast-profile
```

4. Return to privileged EXEC mode.

```
device(config-hardware)# end
```

5. Save the configuration.

```
device# copy running-config startup-config
```

6. Enter the **reload system** command to reboot the device.

```
device# reload system
```

TCAM sharing

Under supported TCAM profiles, you can enable sharing of TCAM resources for each security ACL or PBR ACL applied to multiple ports.



Note

TCAM sharing is supported only on devices based on the DNX chipset family. For a list of such devices, see "Supported Hardware".

No TCAM profile provides simultaneous support for all five flavors of TCAM sharing. The following table displays which and how many TCAM-sharing flavors are supported for each TCAM profile:

Table 9: TCAM-sharing support matrix

TCAM profile	Maximum sharing-flavors	Layer 2 ACL TCAM-sharing	IPv4 ACL TCAM-sharing	IPv4 PBR TCAM-sharing	IPv6 ACL TCAM-sharing	IPv6 PBR TCAM-sharing
default	0	No	No	No	No	No
app-telemetry	0	No	No	No	No	No
border-routing	0	No	No	No	No	No
layer2-ratelimit	0	No	No	No	No	No
multicast-profile	3	Yes	Yes	Yes	No	No
vxlan-visibility	4	Yes	Yes	Yes	Yes	Yes

Enabling TCAM sharing

Follow these steps to enable TCAM sharing.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **hardware** command to enable hardware configuration mode.

```
device(config)# hardware
```

3. Enter the **profile tcam** command to specify the TCAM-sharing profile or profiles that you require.

```
device(config-hardware)# profile tcam cam-share l3-v4-ingress-acl l3-v6-ingress-acl
```

Counter profiles

Counter profiles optimize counters. Common Infrastructure leveraged by applications, to take care of statistics subsystem programming.

Counter Engines (CEs) are used for Application Statistics.

Counting sources are:

- Ingress: InLIF, Ingress PMF(iACL)
- Egress: OutLIF, Egress PMF(eACL)

Counter Profiles determine the amount of CEs each counting source receives.

Each counter profile defines:

- Counter Engine (CE) distribution across counting sources.
- Counter Engine (CE) mode - Hit counter or Forward and Drop Counter.

The following table displays the list of counter profiles.

Table 10: Counter profiles

Profile	Recommended applications
Default	VLAN and BD local switching, MCT
Counter-profile-1	Ingress ACL, OF, Egress ACL
Counter-profile-2	OF, MPLS, VPLS, VLL, MCT
Counter-profile-3	MPLS, VPLS, VLL, MCT
Counter-profile-4	Ingress ACL, VPLS, Egress ACL

Specifying a counter profile

Follow these steps to optimize a specific counter profile.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **hardware** command to enable hardware configuration mode.

```
device(config)# hardware
```

3. Enter the **profile counter** command, specifying a counter profile.

```
device(config-hardware)# profile counters counter-profile-2
```

FIB compression

Border Gateway Protocol (BGP) learns routes from a neighbor and flattens the BGP next-hop route into an IGP next-hop route before downloading those routes to the Routing Information Base (RIB). The RIB then downloads those routes into the Forwarding Information Base (FIB) to program the routes in hardware.

FIB compression is enabled for IPv4 and IPv6, supporting up to (approximately) 5.7 M IPv4 routes and 900 K IPv6 routes. Refer to release notes and scale documentation for further information.

Note the following:

- FIB next-hop/adjacency comprises the set of IGP paths used for forwarding a packet, for example, ECMP paths used by a route to forward matching packets.
- If both a less-specific route and a more-specific route point to the same next-hop/adjacency, the more-specific route is not programmed in hardware.
- The packet for the less-specific route hits the parent route with the same next-hop, ensuring that there is no traffic black hole, and forwarding result is the same.

Compression limitations

Compressions can save hardware resources. However, FIB compression can result in the following:

1. Because compression requires a parent route with the same next-hop, Level 1 routes cannot be compressed, as they may not have a parent route (default route).
2. Level 1 routes in the Internet BGP FIB can be compressed in the range of 40% to 50%.

Configuring FIB compression

Use the following procedure to enable FIB compression on the SLX 9540 and SLX 9640 devices.

1. Enter hardware configuration mode.

```
device# config
device(config)# hardware
device(config-hardware)#
```

2. Enable FIB compression by using the **profile route route-enhance** command as in the following example for both IPv4 and IPv6 compression.

```
device(config-hardware)# profile route route-enhance v4_fib_comp v6_fib_comp on
```

3. Confirm the configuration by using the **show hw route-info** command for an interface.

```
device# show hw route-info interface 1/2
```

```
HW-Route-Info
=====

Slot 1

Tower 0
LEM
Total Entries           :750000
95% Threshold          :712500
85% Threshold          :637500
Total In Use           :39 (.000000%)
    IPV4 routes        :39
    IPV6 routes        :0
Status                 :Green

LPM
Total Entries           :350000
95% Threshold          :332500
85% Threshold          :297500
Total In Use           :331 (.000000%)
    IPV4 routes        :156
    IPV6 routes        :175
Status                 :Green
```

Border profiles for Internet peering

Border profiles for Internet peering supports very highly scaled routing tables. This feature is achieved by using hardware optimization by means of an external TCAM (ETCAM) device for Layer 3 routing, as well as by implementing FIB compression.



Note

This feature is applicable only for SLX 9640 devices.

Previous releases supported Internet routing tables with limited IPv4 routes after FIB compression and hardware optimization features were enabled. This scale is applicable to Internet routing only on the default VRF.

The FIB compression feature compresses route entries to ensure optimal resource utilization. When there is a more-specific and a less-specific route pointing to a same next-hop, FIB compression addresses the more-specific route and programs only the less-specific route in the hardware.

The hardware optimization feature allows the user to program /24 prefix routes in longest exact match (LEM) table. When this feature is enabled, all /23 routes (split into two /24 routes) and /24 prefix routes are programmed into the LEM table. This feature uses more LEM memory than is required for longest prefix match (LPM), and so can be used on devices that have more LEM than LPM capacity.

External TCAM profiles

SLX 9640 devices support up to 5.7 M IPv4 and 900 K IPv6 internet routes under certain ETCAM profiles when FIB compression is enabled. These profiles can be used at border nodes for Internet peering to support Internet route tables over multiple VRFs.

The following external TCAM (ETCAM) profiles are supported, by means of the **profile etcam** in hardware configuration mode:

- Profile ETCAM default: This profile programs IPv4 unicast routes into an external lookup device (ELK), and the internal LPM table is used to program IPv6 unicast routes. This is the default profile in the system.
- Profile ETCAM IPv6-route: This profile programs IPv6 unicast routes into the ELK, and the internal LPM table is used to program IPv4 unicast routes.
- Profile ETCAM IPv4-IPv6-route: This profile programs both IPv4 and IPv6 unicast routes in the ELK.

The following table provides values that can be used for network design purposes.



Important

These values are to be viewed as approximate, for design purposes only. They are based on a compression ratio of 30%. The compression ratio is subject to the routes and next-hop combinations that are available in the system, and it may vary from one network design to another. Refer to release notes and scale documentation for further information.

Table 11: Approximate scale support, per profile, for design purposes

ETCAM profile	FIB compression disabled		FIB compression enabled	
	IPv4 unicast routes	IPv6 unicast routes	IPv4 unicast routes	IPv6 unicast routes
profile etcam default	4,000,000	256,000	5,700,000	365,000
profile etcam ipv6-route	1,000,000	1,000,000	1,400,000	1,400,000
profile etcam ipv4-ipv6-route	4,000,000	700,000	5,700,000	900,000

Configuring support for border profiles

Do the following to configure ETCAM and FIB compression support for border profiles on a SLX 9640.

1. Enter hardware configuration mode.

```
device# config
device(config)# hardware
device(config-hardware)#
```

2. Specify a profile option by using the **profile etcam** command.

```
device(config-hardware)# profile etcam ipv4-ipv6-route
```

This example specifies that IPv4 and IPv6 routes are programmed in the external lookup device (ELK).

3. Enable FIB compression by using the **profile route route-enhance** command, as in the following example for IPv4 and IPv6 routes.

```
device(config-hardware)# profile route route-enhance v4_fib_comp v6_fib_comp on
```

4. Confirm the configuration by using the **show hw route-info** command, as in the following example for a linecard.

```
device# show hw route-info linecard 0

HW-Route-Info
=====

Slot 0

Tower 0
LEM
Total Entries           :750000
95% Threshold          :712500
85% Threshold          :637500
Total In Use           :58 (.000000%)
    IPV4 routes        :58
    IPV6 routes        :0
Status                 :Green

LPM
Total Entries           :1000000
95% Threshold          :950000
85% Threshold          :850000
Total In Use           :696 (.000000%)
    IPV4 routes        :0
    IPV6 routes        :174
Status                 :Green

eTCAM
Total Entries           :4000000
95% Threshold          :3800000
85% Threshold          :3400000
Total In Use           :156 (.000000%)
    IPV4 routes        :156
    IPV6 routes        :0
Status                 :Green
```

Hardware profile show commands

There are several show commands that display hardware-profile information, as listed in the following table.

Table 12: Hardware profile show commands in the *Command Reference*

Command	Description
show hardware profile	Displays details of the all current hardware profiles. You can also display TCAM-sharing details, or details for a specific TCAM, counter, or LAG profile.
show hw route-info	Displays the route-info counters.

Enter Maintenance Mode Before Performing Device Maintenance

Maintenance mode helps to minimize traffic loss during planned maintenance operations such as software upgrade, SFP replacement, cable replacement, and node replacement.

Planned maintenance operations may require the device to be shut down or restarted, resulting in traffic disruption even if alternative paths are available. Maintenance mode provides graceful traffic diversion to alternative traffic paths, helping to minimize traffic loss during such planned operations.

When an alternative path is available, the BGP and MCT protocols redirect traffic away from the node that is going into maintenance mode. When maintenance mode is enabled, all protocols that are running on the maintenance mode node are notified and redirection of traffic (convergence) begins in stages.



Note

Maintenance mode is not supported for the following features: BGP address-family, Flowspec, Layer 3 VPN, VPLS, and VLL (virtual leased line).

1. Access configuration mode.

```
device# configure terminal
```

2. Access system mode.

```
device(config)# system
```

3. Access system maintenance mode.

```
device(config-system)# maintenance
```

4. Enable maintenance mode.

```
device(config-system-maintenance)# enable
```

5. Specify the number of seconds allowed per stage of the convergence of traffic to the maintenance mode node.

```
device(config-system-maintenance)# convergence-time 125
```

This example sets the convergence time to 125 seconds.

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# system
device(config-system)# maintenance
```

```
device(config-system-maintenance)# enable  
device(config-system-maintenance)# convergence-time 125
```



SLX-OS and Linux Shell Interoperability

[Overview on page 70](#)

[Executing Linux shell commands from SLX-OS on page 71](#)

[Executing scripts from SLX-OS on page 72](#)

[Accessing the Linux shell from SLX-OS on page 73](#)

[Executing SLX-OS commands from the Linux shell on page 73](#)

[Escalating Linux permissions to root on page 74](#)

[Saving and appending show command output to a file on page 75](#)

[Logs of Linux shell activities on page 75](#)

Overview

The SLX-OS supports interoperability between the SLX-OS CLI and the SLXVM Linux shell.

As an SLX-OS user with admin permissions, you can perform the following tasks:

- Running permitted Linux commands and scripts from the SLX-OS CLI
- Accessing the SLXVM Linux shell, and:
 - Running permitted Linux commands and scripts. However, if you have access to the root password, you can then escalate your permissions, by using the **su root** Linux command.
 - Running SLX-OS configuration and show commands.
 - Running scripts that contain multiple SLX-OS commands.

Limitations

- By default, only the Bash shell is supported. With Linux root permissions, you can install a different shell, such as the C shell or KornShell. However, shell-activity logging is supported only for the Bash shell.
- If you open multiple Bash sessions, the Linux shell timeout is applicable only on the current Bash session.
- If you run Linux commands as part of the script or through a file, the device logs the script or file execution. It does not log the commands.
- If you use the **cli_run** command to execute SLX-OS CLI **show** commands from the shell, pagination is not supported, and commands that require user input are also not supported.
- At the SLX-OS CLI, a window resizing issue occurs when you execute Linux commands such as **top** using the **oscmd** command. Extreme recommends that you execute these commands from the Linux shell.

- Although as an SLX-OS admin, you have permissions to run the following commands from the Linux shell, you do not have permissions to run them—from the SLX-OS CLI—appended to the **oscmd** command.
 - **bash**
 - **script**
 - **vi**
 - **vim**
- Do not modify SLX-OS user accounts from the Linux shell. For information on modifying user accounts, refer to the *Extreme SLX-OS Security Configuration Guide*.

Executing Linux shell commands from SLX-OS

You can execute a Linux command from the SLX-OS CLI, appended to **oscmd**.

In the following example, the Linux **ps -ef** command lists the process status.

```
device# oscmd ps -ef
UID      PID  PPID  C  STIME TTY          TIME CMD
root      1    0    0 Jul24 ?           00:00:04 /sbin/init
root      2    0    0 Jul24 ?           00:00:00 [kthreadd]
root      3    2    0 Jul24 ?           00:00:00 [migration/0]
root      4    2    0 Jul24 ?           00:00:03 [ksoftirqd/0]
root      5    2    0 Jul24 ?           00:00:00 [migration/1]
root      6    2    0 Jul24 ?           00:00:03 [ksoftirqd/1]
root      7    2    0 Jul24 ?           00:00:00 [migration/2]
root      8    2    0 Jul24 ?           00:00:02 [ksoftirqd/2]
root      9    2    0 Jul24 ?           00:00:00 [migration/3]
root     10    2    0 Jul24 ?           00:00:02 [ksoftirqd/3]
root     11    2    0 Jul24 ?           00:00:00 [migration/4]
root     12    2    0 Jul24 ?           00:00:02 [ksoftirqd/4]
root     13    2    0 Jul24 ?           00:00:00 [migration/5]
root     14    2    0 Jul24 ?           00:00:03 [ksoftirqd/5]
root     27    2    0 Jul24 ?           00:00:00 [cpuset]
root     28    2    0 Jul24 ?           00:00:01 [khelper]
root     31    2    0 Jul24 ?           00:00:00 [netns]
root     34    2    0 Jul24 ?           00:00:00 [async/mgr]
root    270    2    0 Jul24 ?           00:00:00 [sync_supers]
root    272    2    0 Jul24 ?           00:00:00 [bdi-default]

...

root      8kblockd/6]182    1  0 Jul24 ?           00:00:00 /usr/sbin/inetd
root     8237    1  0 Jul24 ?           00:00:00 /usr/sbin/sshd
admin   27536 27535  0 04:19 pts/4           00:00:00 ps -ef
```

Executing scripts from SLX-OS

From the SLX-OS CLI, you can execute scripts that you copied to flash memory or created in the SLXVM Linux shell.

Downloading a script to the SLX-OS device

After writing and testing a user-defined script file, copy it from an accessible network location to the flash memory of the SLX-OS device.

```
device# copy ftp://MyUserID:MyPassword@10.10.10.10//<copy_script.sh> flash://  
copy_script.sh
```

After copying the script to the device, verify that the script file is displayed with the list of files in the flash memory of the device.

```
device# dir  
total 24  
drwxr-xr-x  2 root    sys      4096 Oct 26 15:22 .  
drwxr-xr-x  3 root    root     4096 Oct  1 1970 ..  
-rw-r--r--  1 root    root     1051 Oct 24 16:09 copy_script.sh  
-rw-r--r--  1 root    root      207 Oct 24 16:09 create_vlans.py  
-rw-r--r--  1 root    sys       557 Oct 26 10:37 defaultconfig.novcs  
-rw-r--r--  1 root    sys       778 Oct 26 10:37 defaultconfig.vcs  
  
1922789376 bytes total (828317696 bytes free)
```

If the copied script does not have executable permissions, you need to assign executable permissions from the SLXVM Linux shell. Note that you need root access for this action, as described in "Escalating Linux permissions to root."

```
[root@SLX]# cd /var/config/vcs/scripts/  
[root@SLX]# chmod 755 copy_script.sh  
[root@SLX]# ls -lart copy_script.sh  
-rwxr-xr-x 1 root root 1051 Oct 24 16:09 copy_script.sh
```

You can also display the contents of a script file.

```
device# show file copy_script.sh
```

Creating scripts in the Linux shell

You can create scripts by using the Linux shell **vi** editor, as in the following example.

```
device# start-shell  
Entering Linux shell for the user: admUser  
[admUser@SLX]# cd scripts  
[admUser@SLX]# vi create_script.sh
```

After you write the script, make sure that it exists in the `/fabos/users/admin/script` directory and is executable under Linux.

```
[admUser@SLX]# pwd  
/fabos/users/admin/scripts
```


Running scripts from the SLX-OS CLI

You can run scripts directly from SLX-OS CLI. Enter **oscmd** followed by the name of the script.

```
device# oscmd my_script
```

Accessing the Linux shell from SLX-OS

With admin-level permissions, you can access the SLXVM Linux shell from the SLX-OS CLI, by using the **start-shell** command.



Note

Inside the SLXVM Linux shell, you can execute commands that do not require root permissions. To escalate your permissions, refer to "Escalating Linux permissions to root".

1. To access the SLXVM Linux shell, enter the **start-shell** command.

```
device# start-shell
Entering Linux shell for the user: admUser
[admUser@SLX]#
```

2. Enter Linux commands and run scripts as needed. You can also run SLX-OS commands from the Linux shell.
3. To exit the shell and return to the SLX-OS CLI, enter **exit**.

```
[admUser@SLX]# exit
```

Upon exiting, the following message appears and you return to the SLX-OS CLI prompt.

```
exit
Exited from Linux shell
device#
```

Executing SLX-OS commands from the Linux shell

From the SLXVM Linux shell, you can execute a single SLX-OS command or a file that contains a series of such commands.

1. To execute an SLX-OS command, enter the Linux **cli_run -c** command.

```
[admUser@SLX]# cli_run -c "show ip interface brief" | grep Port-channel > /tmp/
interface
```

In the previous example, the output of **show ip interface brief** is redirected to the `/tmp/interface` file.

2. Display the contents of the file to verify the redirection.

```
[admUser@SLX]# cat /tmp/interface
Port-channel 1      unassigned      administratively down    down
Port-channel 2      unassigned      administratively down    down
```

3. To execute a file containing multiple SLX-OS commands, enter the Linux **cli_run -f** command.

```
[admUser@SLX]# cli_run -f /tmp/slxccli_cmd_file > /tmp/newfile
```

In this example, `slxccli_cmd_file` contains the following commands:

```
[admUser@SLX]# cat /tmp/slxccli_cmd_file
show ssh server status
conf t
router bgp
```

```
local-as 23
capability as4-enable
```

**Note**

Make sure that each command is on a new line.

4. Display the contents of the target file to verify that it contains the redirected output.

```
[admUser@SLX]# cat /tmp/newfile
Welcome to the Extreme SLX-OS Software
admin connected from 127.0.0.1 using console on SLX
SLX# show ssh server status | nomore
VRF-Name: mgmt-vrf      Status: Enabled
VRF-Name: default-vrf  Status: Enabled
device# conf t
Entering configuration mode terminal
Current configuration users:
admin console (cli from 10.70.4.183) on since 2017-01-31 05:49:59 terminal mode
device(config)# router bgp
device(config-bgp-router)# local-as 23
device(config-bgp-router)# capability as4-enable
device(config-bgp-router)#
```

Escalating Linux permissions to root

In the SLXVM Linux shell, you can escalate your default admin access to root access (password protected).

**Caution**

A user with SLXVM Linux-shell root permissions can—unintentionally or maliciously—execute commands that can render the SLX inoperable.

1. From the SLX-OS CLI prompt, enter **start-shell** to access the SLXVM Linux shell.

```
device# start-shell
Entering Linux shell for the user: admUser
[admUser@SLX]#
```

You can now execute commands that do not require root permissions.

2. To escalate your permissions, enter the Linux **su root** command.

```
[admUser@SLX]# su root
Password:
```

3. Enter the root password.

```
Password:
```

After successful login, the following warning is displayed:

```
We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

#1) Respect the privacy of others.
#2) Think before you type.
#3) With great power comes great responsibility.
[root@SLX]#
```

4. Enter Linux commands and run scripts as needed.

You can also run SLX-OS commands from the Linux shell.

- To exit root level and return to the default SLXVM Linux shell, enter **exit**.

```
[root@SLX]# exit
exit
[admUser@SLX]#
```

- To exit the default SLXVM Linux shell and return to the SLX-OS CLI, enter **exit**.

```
[admUser@SLX]# exit
exit
Exited from Linux shell
device#
```

Saving and appending show command output to a file

For output of a **show** command saved or appended to a file, the **oscmd** command enables you to display the file.

- Save the **show** command output to a file.

```
device# show ssh server status | save status
```

In this example, the **show ssh server status** output is saved to the status file.

- Display the contents of the file with the Linux **cat** command.

```
device# oscmd cat status
VRF-Name: mgmt-vrf      Status: Enabled
VRF-Name: default-vrf  Status: Enabled
```

- Append the **show** output to an existing file.

```
device# show ip interface brief | last 5 | append status
```

In this example, the **show ip interface brief** output is appended to the status file.

- Display the contents of the file with the Linux **cat** command.

```
device# oscmd cat status
VRF-Name: mgmt-vrf      Status: Enabled
VRF-Name: default-vrf  Status: Enabled
Ethernet 2/58           unassigned  default-vrf  administratively down  down
Ethernet 2/59           unassigned  default-vrf  administratively down  down
Ethernet 2/60           unassigned  default-vrf  administratively down  down
Ethernet 2/125(I)       unassigned  default-vrf  administratively down  down
Ethernet 2/126(I)       unassigned  default-vrf  administratively down  down
```

Logs of Linux shell activities

By default, SLX-OS logs users entering the SLXVM Linux shell, commands executed in that shell, and users exiting from the Linux shell back to the SLX-OS CLI.

Linux shell user entry and exit logs

SLX-OS uses RASLOG to log entries when the user enters and exits the Linux shell. If you configure a remote Syslog server, the same logs can also be seen on that server.

From privileged EXEC mode, use the **show logging raslog** command to display the RASLOG entries.

- When a user enters the Linux shell, the **show logging raslog** command displays an SH-1001 message.

```
device# show logging raslog
```

```
2016/06/25-06:42:54, [SH-1001], 1547, M1 | Active, INFO, SLX, SLXVM Linux shell login
information: User [admUser]. Login Time : Sat Jun 25 06:42:54 2016
```

- When a user exits the Linux shell, the **show logging raslog** command displays an SH-1002 message.

```
device# show logging raslog
```

```
2016/06/25-06:43:59, [SH-1002], 1548, M1 | Active, INFO, SLX, Event: exit, Status:
success, Info: User [admUser] successfully exited from SLXVM Linux shell. Exit Time:
Sat Jun 25 06:43:59 2016
```



Note

An SH-1003 message indicates failure to log in to the Linux shell.

Linux shell command execution logs

Command activities at the Linux shell are logged locally in the `/var/log/shell_activity.log` file and remotely on a Syslog server.

When a user executes a command at the Linux shell, the `shell_activity.log` file includes SH-1005 messages:

```
[admUser@SLX]# tail -f /var/log/shell_activity.log
```

```
shell: [log@1588 value="SHELL"][timestamp@1588 value="2017-12-14T11:17:03"][msgid@1588
value="SH-1005"][severity@1588 value="INFO"][swname@1588 value="SLX9540"][arg0@1588
value="no" desc="root access"][arg1@1588 value="admin" desc="username"] BOM Executed
command at Linux shell : pwd
shell: [log@1588 value="SHELL"][timestamp@1588 value="2017-12-14T11:17:18"][msgid@1588
value="SH-1005"][severity@1588 value="INFO"][swname@1588 value="SLX9540"][arg0@1588
value="no" desc="root access"][arg1@1588 value="admin" desc="username"] BOM Executed
command at Linux shell : ls
```



Note

The `/var/log/shell_activity.log` file is rotated every thirty minutes if it goes over 2 MB in size. The old version of the file is compressed; a maximum of four rotated files can exist at the same time.

Configuring remote logging of Linux shell activities

By default, SLX-OS logs Linux shell commands both locally (in `/var/log/shell_activity.log`) and remotely, on the Syslog server.

From SLX-OS CLI, you can perform the following tasks to control the logging of commands executed at the Linux shell to a remote Syslog server. These tasks do not affect the local logging.



Note

Changes of the **log-shell stop** and **log-shell start** commands are applicable only on new Linux shell sessions.

1. To disable remote logging, enter **log-shell stop**.

```
device# log-shell stop
```

Local logging of user activities continues.

2. To restart remote logging, enter **log-shell start**.

```
device# log-shell start
```

3. To check the remote logging status, enter **log-shell status**.

```
device# log-shell status
```

When remote logging is enabled, the following message is displayed.

```
Linux shell activity logging : Enabled
```



Guest OS for TPVM

[VM Access Management](#) on page 78

[Insight Interface and TPVM](#) on page 83

[TPVM](#) on page 97

TPVM, or Third-Party Virtual Machine, is a general server that resides on ExtremeSLX-OS devices. The guest OS that it provides is different from SLX-OS.

VM Access Management

This section addresses how the SLX-OS accesses the Third-Party Virtual Machine (TPVM).

Extreme SLX-OS devices support the provisioning of a Guest OS, referred to as TPVM or the Third-Party Virtual Machine. Currently only one instance of the TPVM and one image for the TPVM is provided. The Extreme SLX-OS will run the TPVM in the following modes; VM Mode on the SLX 9540, and Baremetal on the SLX 9640, SLX 9150, and SLX 9250. The figure below describes the two modes and how multiple OS's are stacked, followed by a table for platform support details.

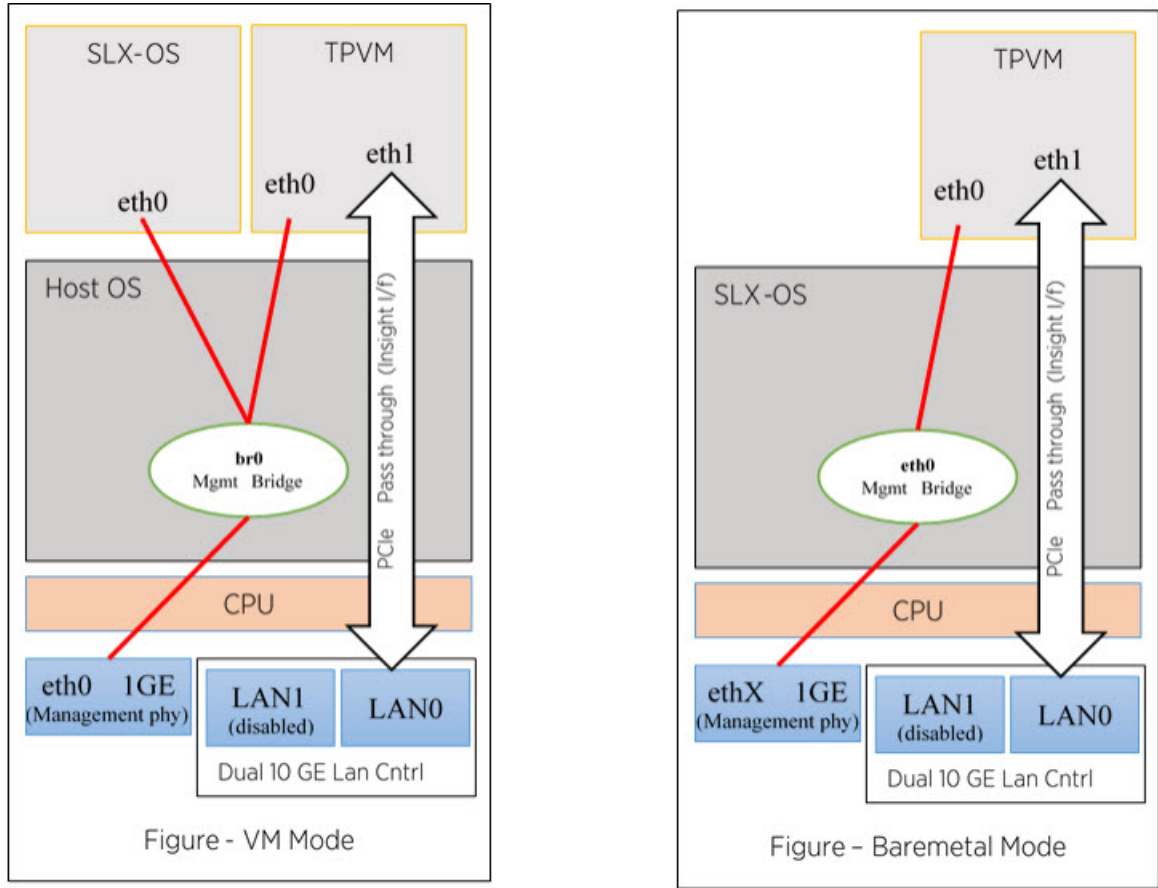


Figure 8: SLX-OS VM and Baremetal Modes

Table 13: Platform OS Mode Details

Platform	OS Mode	Host OS Image Version	SLX-OS Image Version	Image	TVPM Image Version	Insight Interface
SLX 9540	VM Mode	Ubuntu 14.04	Linux 4.14.67	SLX	Ubuntu 16.04.4 LTS	Yes
SLX 9640	Baremetal	N/A	Linux 4.14.67	SLX	Ubuntu 16.04.4 LTS	Yes
SLX 9150	Baremetal	N/A	Linux 4.14.67	SLX	Ubuntu 16.04.4 LTS	Yes
SLX 9250	Baremetal	N/A	Linux 4.14.67	SLX	Ubuntu 16.04.4 LTS	Yes

Extreme SLX-OS VM Access Management

Each SLX platform has data plane access for Third-Party Virtual Machine (TPVM) applications through the Insight Interface.

On the SLX 9540, the front-panel port 0/48 is shared with the insight interface (port 0/125) through a command line controlled hardware switch. This interface can only be operational as either a data

forwarding port, or as an insight interface at any given time. By default, interface 0/48 is operational as a data forwarding port. When insight mode is configured, interface 0/48 is deleted dynamically along with any associated configurations, and interface 0/125 is created.

On the SLX 9640, the front-panel port 0/24 is shared with the insight interface (port 0/126) through a command line controlled hardware switch. This interface can only be operational as either a data forwarding port, or as an insight interface at any given time. By default, interface 0/24 is operational as a data forwarding port. When insight mode is configured, interface 0/24 is deleted dynamically along with any associated configurations, and interface 0/126 is created.

The SLX 9150 and SLX 9250 assign the following ports for VM access;

- SLX 9150T : 0/73
- SLX 9150: 0/81
- SLX 9250: 0/129

Default credentials for the hosts

The following table provides the default user credentials to access the hosts of the operating systems on SLXVM1 and the commands to change the passwords.

Table 14: Default credentials for the hosts

Host name	Default user login	Default password	Changing the password
SLX-OS	root	fibranne	From the SLXVM shell, the passwd command
	admin	password	From the SLXVM CLI, the username command with RBAC rules
	user	password	
HOST-OS	root	fibranne	From the SLXVM shell, use the hostadm update_passwd command
TPVM	admin	password	From the SLX CLI, run tpvm password

VM access

You can access the VMs and host operating systems through the SLX-OS CLI, SLXVM OS shell or the serial console.

SLX-OS CLI to VM access

The SLX-OS CLI allows you to access to the SLXVM shell by using the **start-shell** command.

When you log into the SLX-OS CLI, you can use the default admin or user credentials. Non-default users are authenticated through AAA.

The following example shows Telnet access to the SLX-OS CLI with admin credentials.

```
client# telnet 10.24.12.71
Trying 10.24.12.71...
Connected to 10.24.12.71.
Escape character is '^]'.
SLX login: admin
```



```

Password:
SECURITY WARNING: The default password for at least
one default account (root, admin and user) have not been changed.

Welcome to the Extreme SLX Operating System Software
admin connected from 10.70.5.113 using telnet on device
device#

```

The default SLX-OS prompt is SLX#. However, throughout this guide, the device# is used as the prompt.

To exit the session, enter **exit**.

Serial Console Access

When using the SLX 9540 in VM Mode, the shortcut keys shown below allow access to the SLXVM, Host OS, or TVPM operating systems.

For SLX platforms operating in Baremetal Mode (SLX 9640, SLX 9150), use the `tpvm console` command to access the TVPM serial console port, and the key sequence **Ctrl+\<** to return to the SLX console.

Table 15: Shortcut keys

Operating system	Shortcut keys
Host OS	Ctrl+Y+1
SLXVM OS	Ctrl+Y+2
TPVM	Ctrl+Y+3

SLX 9540 Serial Console to Host OS example

```

device# Ctrl+Y+1
Ubuntu 14.04 LTS HOST ttyS0

HOST login: root
Password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.14.17 x86_64)
...
...
root@HOST:~#

```

SLX 9540 Serial Console to SLXVM Shell example

```

device# Ctrl+Y+2
[admin@SLX]#

```

SLX 9540 Serial Console to TPVM example

From the SLXVM shell, start TPVM.

```

[admin@SLX]# tpvm install
Installation starts. To check the status, run 'tpvmadm show'
[admin@SLX]# show tpvm status
TPVM is installed but not running, and AutoStart is disabled on this host.
[admin@SLX]# tpvm start

```

```
start succeeds
[admin@SLX]#
```

Authenticate and access the TPVM shell prompt.

```
[admin@SLX]# Ctrl+Y+3
TPVM login: root
Password:
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.19.0-25-generic x86_64)
...
root@TPVM:~#
root@TPVM:~# exit

Ubuntu 14.04.3 LTS TPVM ttyS0

TPVM login:
```

To return to the SLXVM CLI shell prompt, enter **Ctrl+Y+2**.

Accessing the Linux shell from SLX-OS

With admin-level permissions, you can access the SLXVM Linux shell from the SLX-OS CLI, by using the **start-shell** command.



Note

Inside the SLXVM Linux shell, you can execute commands that do not require root permissions. To escalate your permissions, refer to "Escalating Linux permissions to root".

1. To access the SLXVM Linux shell, enter the **start-shell** command.

```
device# start-shell
Entering Linux shell for the user: admUser
[admUser@SLX]#
```

2. Enter Linux commands and run scripts as needed. You can also run SLX-OS commands from the Linux shell.
3. To exit the shell and return to the SLX-OS CLI, enter **exit**.

```
[admUser@SLX]# exit
```

Upon exiting, the following message appears and you return to the SLX-OS CLI prompt.

```
exit
Exited from Linux shell
device#
```

Accessing the SLXVM1 Host-OS shell from the SLX-OS CLI

You can access the SLXVM1 host-OS shell from the SLX-OS CLI from an SSH session.

In the following example, an SSH session is used to access the SLXVM1 host-OS shell.

```
device# ssh SLXVM1_kvm vrf mgmt-vrf -l root
The authenticity of host 'SLXVM1_kvm (127.2.0.1)' can't be established.
ECDSA key fingerprint is d5:ba:cc:d4:57:03:e4:b2:6f:ca:d2:dd:4c:40:5d:60.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'SLXVM1_kvm' (ECDSA) to the list of known hosts.
root@SLXVM1_kvm's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.14.17 x86_64)
 * Documentation:  https://help.ubuntu.com/
System information as of Thu Aug  4 10:41:47 PDT 2016
System load:  0.81           Users logged in:  0
Usage of /:   14.7% of 9.72GB IP address for eth1: 127.2.0.1
Memory usage: 54%           IP address for br0: 10.24.12.77
```

```

Swap usage:    0%                IP address for virbr0: 192.168.122.1
Processes:    164
Graph this data and manage this system at:
  https://landscape.canonical.com/
Last login: Thu Aug  4 10:41:47 2016 from cp0_vm1
root@HOST:~#

```

In the previous examples, use **exit** to terminate the session and return back to the parent shell.

Serial console to VM access

The serial console allows access to the following:

- SLXVM OS
- TPVM

Initial authentication occurs at the SLX-OS CLI prompt.

```

SLX login: admin
Password:

SECURITY WARNING: The default password for at least
one default account (root, admin and user) have not been changed.

Welcome to the Extreme SLX Operating System Software
admin connected from 127.0.0.1 using console on SLX
device#

```

Each OS requires its own credentials.

TPVM address assignment

TPVM has Ethernet 0 and 1 interfaces. Ethernet 0 (eth0) is enabled and its IP address is configured using DHCP.

On the SLXVM, the **show tpvm ipaddr** command configures the address.

If DHCP is not available, you can use the serial console to log in as root, and use standard Ubuntu (14.04 LTS) commands to configure the IP address for eth0 interface.

Eth1 is connected to the Insight interface and allows packet monitoring.

Insight Interface and TPVM

The Insight Interface is the port that provides data plane access for Third-Party Virtual Machine (TPVM) applications.

TPVM is a server that resides on Extreme SLX-OS devices, connected through the Insight Interface. It may be used in one of the following modes:

- Data plane traffic mirroring - Analytic mode
- TPVM Reachability - Bi-directional Reachability mode

Support for TPVM is through a front-panel port (SLX 9540 and SLX 9640), shared with the Insight Interface using a hardware switch by means of a CLI command. The SLX 9150 and SLX 9250 have a dedicated Insight port. Physically, it may be a direct or indirect ethernet point-to-point connection between the device fast forwarding Data Plane ASIC Chip port to the TPVM. In order to use TPVM, each

endpoint must be set up individually on the appropriate OS (the SLX-OS and the TVPM OS). The following table details access to the Insight Interface on Extreme SLX platforms.

Table 16: Insight Interface Support on Extreme SLX platforms

Extreme SLX device	Support for insight interface
SLX 9540	On port 0/48 (Insight interface 0/125 is created)
SLX-9640	On port 0/24 (Insight interface 0/125 is created)
SLX-9250	Dedicated Insight port
SLX-9150-48XT	Dedicated Insight port
SLX-9150-48Y	Dedicated Insight port

The Insight Interface endpoint is configured from the command line as a Port Channel with Insight enabled. There can only be one such Port Channel on the device; you cannot add any new members to this Port channel. However, a port channel with existing members can have Insight enabled.

On the TPVM, the Insight Interface endpoint shows as Linux Network Interface **eth1**, and is configured statically. For DHCP-based configuration, refer to the section below

The following section addresses the management details of using the Insight Interface port on supported Extreme SLX devices. For the details of TPVM applications supported on all Extreme SLX devices, refer to "TPVM" later in this chapter.

Insight interface port-channel

This section addresses details of the insight interface port on different Extreme SLX platforms.

Extreme SLX 9540

- Either front-port 0/48 or the insight interface port 0/125 can exist at a given time, as these ports share the same ASIC port.
- By default, 0/48 is created.
- Insight mode can be configured by means of the **connector 0/48** command. A port-channel cannot be made an insight port-channel until insight interface mode is enabled on connector 0/48. Similarly, insight mode cannot be removed on connector 0/48 until the connector is unbound from the insight port-channel.
- Upon insight mode configuration, interface 0/48 is dynamically deleted and 0/125 is created.
- All the configurations under interface 0/48 are deleted upon insight mode configuration.
- An existing port-channel with existing member ports cannot be made an insight interface port-channel.

The following is an example configuration.

```
device(config)# hardware
device(config-hardware)#
device(config-hardware)# connector 0/48
device(config-connector-0/48)# [no] insight mode
device(config)# interface port-channel 20
device(config-Port-channel-20)# insight enable
```

Extreme SLX 9640

- Either front-port 0/24 or the insight interface port 0/126 can exist at a given time, as these ports share the same ASIC port.
- By default, 0/24 is created.
- Insight mode can be configured by means of the **connector 0/24** command. A port-channel cannot be made an insight port-channel until insight interface mode is enabled on connector 0/24. Similarly, insight mode cannot be removed on connector 0/24 until the connector is unbound from the insight port-channel.
- Upon insight mode configuration, interface 0/24 is dynamically deleted and 0/126 is created.
- All the configurations under interface 0/24 are deleted upon insight mode configuration.
- An existing port-channel with existing member ports cannot be made an insight interface port-channel.

Extreme SLX 9150 and SLX 9250

The SLX 9150 and SLX 9250 assign the following ports for VM access. These ports are exclusive use only for VM access.

- SLX 9150T : 0/73
- SLX 9150: 0/81
- SLX 9250: 0/129

Insight interface

Insight interface supported features

Insight interface port-channel supports the following third-party features and hardware.

Insight interface supports the standard features seen in other front end interfaces including:

- Port and ACL-based mirroring destination
- QoS and rate shaping

Insight Interface Data Path

The Insight Interface port is provided as a pre-existing port-channel interface to which all the Insight Interfaces are automatically added during system boot.

The port-channel interface is created as default LAGs in the system. It is visible to you, configured with default settings, and is a static LAG. All other options on the LAG are disabled. Insight Interface ports and port-channels work independently with each providing up to 20 GB bandwidth for applications.

On the SLX 9540, port 0/48 is multiplexed for normal use and as access to the Insight Interface. On the SLX 9640, port 0/24 is used.

When Insight is invoked, the hardware switch reconfigures port 0/48 (or 0/24 on the SLX 9640) to ethernet interface 0/125 and is used exclusively for Insight configuration/management. The user never has to specifically configure Eth 0/125. When a **show** command is run, no configuration is displayed for port 0/48 (or 0/24 on the SLX 9640).

Insight Interface port-channel creation, addition, or deletion is similar to the standard port-channel creation, addition, or deletion except it is programmatically invoked during system initialization.

Configure the Insight Interface SLX Endpoint

The Insight Interface is configured on both the SLX endpoint and the TPVM endpoint. Use the following procedure to configure the SLX endpoint of the Insight Interface. Note that steps 1 through 4 are specific to the SLX 9540 and SLX 9640.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **hardware** command to enter hardware configuration mode. (This is required for the SLX 9540/9640 only.)

```
device(config)# hardware
```

3. Enter the **connector** command to specify a slot and port.

```
device(config-hardware)# connector 0/48
```

4. Enter the **insight mode** command to enable the insight interface on a port-channel, and exit to global configuration mode.

```
device(connector-0/48)# insight mode
```

For the SLX 9640 use Eth 0/24. The command will toggle the physical port, and will display as port 0/125 in any **show** commands.

5. Exit to global configuration mode.

```
device(connector-0/48)# exit
```

6. In global configuration mode, specify a port-channel. You can create a new port-channel or use an existing unconfigured port-channel. The Insight Interface can only be configured on one port-channel

```
device(config)# interface port-channel 22
```

7. In interface subtype configuration mode, enter the **insight enable** command.

```
device(config-Port-channel-22)# insight enable
```

8. Set the port ip-address.

```
device(config-Port-channel-20)# ip address 10.0.0.1/24
```

9. Enable the interface.

```
device(config-Port-channel-22)# no shutdown
```

10. Use the **show interface port-channel** and the **show port-channel** commands to confirm the configuration, as in the following example.

```
device# show interface port-channel 22
Port-channel 22 is up, line protocol is up
Hardware is AGGREGATE, address is 609c.9f5a.4558
  Current address is 609c.9f5a.4558
Interface index (ifindex) is 671088673
Minimum number of links to bring Port-channel up is 1
MTU 1548 bytes
LineSpeed Actual      : 10000 Mbit
Allowed Member Speed : 10000 Mbit
Priority Tag disable
Forward LACP PDU: Disable
Route Only: Disabled
Last clearing of show interface counters: 1d23h53m
Queueing strategy: fifo
```

```

Receive Statistics:
  0 packets, 0 bytes
  Unicasts: 0, Multicasts: 0, Broadcasts: 0
  64-byte pkts: 0, Over 64-byte pkts: 0, Over 127-byte pkts: 0
  Over 255-byte pkts: 0, Over 511-byte pkts: 0, Over 1023-byte pkts: 0
  Over 1518-byte pkts(Jumbo): 0
  Runts: 0, Jabbers: 0, CRC: 0, Overruns: 0
  Errors: 0, Discards: 0
Transmit Statistics:
  5 packets, 380 bytes
  Unicasts: 0, Multicasts: 5, Broadcasts: 0
  Underruns: 0
  Errors: 0, Discards: 0
Rate info:
  Input 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
  Output 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
Route-Only Packets Dropped: 0
Time since last interface status change: 00:00:21

device# show port-channel 22
Static Aggregator: Po 22
Aggregator type: Standard
Number of Ports: 1
Member ports:
  Eth 0/125  *

```

Configure the Insight Interface TVPM Endpoint for Analytic Mirroring

Use the following procedures to configure the Insight Interface TVPM endpoint for Analytic Mirroring. The following procedure assumes that the TVPM license is installed and active, and that TVPM is installed. Use Linux commands to configure a Layer 3 IP address and route for the ethernet interface.

1. Start TVPM.

```
device# tvpm start
```

2. From a LINUX prompt, configure Eth1 to promiscuous mode.

```
bash# ifconfig eth1 promiscuous
bash# tcpdump -i eth1
```

3. In global configuration mode, specify a port-channel. You can create a new port-channel or use an existing unconfigured port-channel. The Insight Interface can only be configured on one port-channel

```
device(config)# interface port-channel 20
```

4. In interface subtype configuration mode, enter the **insight enable** command.

```
device(config-Port-channel-20)# insight enable
```

5. Enable the interface.

```
device(config-Port-channel-20)# no shutdown
```

6. From the SLX CLI, configure the session monitor.

```
device(config)# monitor session 1
device (config-session-1)# source ethernet 0/49 destination port-channel 20
direction both
```

Configure the Insight Interface TVPM Endpoint for Routing

Use the following procedures to configure the Insight Interface TVPM endpoint for Bi-directional Routing. The following procedure assumes that the TVPM license is installed and active, and that TVPM is installed. Use Linux commands to configure a Layer 3 IP address and route for the ethernet interface.

1. Start TVPM.

```
device# tvpm start
```

2. From a LINUX prompt, configure the IPV4 address and route entry.

```
bash# ifconfig eth1 10.0.0.100 netmask 255.255.255.0
bash# route add -net 1.1.1.0 netmask 255.255.255.0 gw 10.0.0.1
```

3. In global configuration mode, specify a port-channel. You can create a new port-channel or use an existing unconfigured port-channel. The Insight Interface can only be configured on one port-channel

```
device(config)# interface port-channel 20
```

4. In interface subtype configuration mode, enter the **insight enable** command.

```
device(config-Port-channel-20)# insight enable
```

5. Set the port ip-address.

```
device(config-Port-channel-20)# ip address 10.0.0.1/24
```

6. Enable the interface.

```
device(config-Port-channel-20)# no shutdown
```

Configure the Insight Interface TVPM Endpoint for Switching

Use the following procedures to configure the Insight Interface TVPM endpoint for Switching. The following procedure assumes that the TVPM license is installed and active, and that TVPM is installed. Use Linux commands to configure a Layer 3 IP address and route for the ethernet interface.

1. Start TVPM.

```
device# tvpm start
```

2. From a LINUX prompt, configure the IPV4 address and route entry.

```
bash# ifconfig eth1 10.0.0.100 netmask 255.255.255.0
bash# route add -net 1.1.1.0 netmask 255.255.255.0 gw 10.0.0.1
```

3. In global configuration mode, specify a port-channel. You can create a new port-channel or use an existing unconfigured port-channel. The Insight Interface can only be configured on one port-channel

```
device(config)# interface port-channel 20
```

4. In interface subtype configuration mode, enter the **insight enable** command.

```
device(config-Port-channel-20)# insight enable
```

5. Perform the following configuration:

```
SLX# conf t
Entering configuration mode terminal
SLX(config)# vlan 100
SLX(config-vlan-100)# router-interface ve 100
SLX(config-vlan-100)# exit
SLX(config)# interface ve 100
SLX(config-if-Ve-100)# ip address 10.0.0.1/24
SLX(config-if-Ve-100)# no shut
SLX(config-if-Ve-100)# exit
```



```
SLX(config)# interface port-channel 20
SLX(config-Port-channel-20)# switchport
SLX(config-Port-channel-20)# switchport mode access
SLX(config-Port-channel-20)# switchport access vlan 100
SLX(config-Port-channel-20)# no shutdown
```

Inbound ACL-based mirroring

A Layer 2 or Layer 3 extended ACL permit filter must be configured to mirror the incoming matching traffic to a given port. You can also configure different mirror ports for different filters in the same ACL.

Enabling ACL-based port mirroring

Follow these high level steps to enable ACL-based port mirroring.

1. Create an ACL.
 - Traffic can only be selected using a permit clause.
 - The ACL can be bound to a physical port or a LAG.
 - The physical port or LAG interface should be configured as a switchport.
 - Configure the **mirror** keyword in an ACL filter to enable inbound ACL mirroring. This directs selected traffic to the mirrored port.
2. Associate the ACL mirror source and destination port. The mirror source port should be physical and the mirror destination port is either a physical port or a LAG port.
3. Bind the ACL to an interface.
4. Save the configuration.

Related Links

[Configuring inbound ACL-based mirroring to the insight interface](#) on page 89

Follow these steps to configure inbound ACL-based mirroring.

Configuring inbound ACL-based mirroring to the insight interface

Follow these steps to configure inbound ACL-based mirroring.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create an extended Layer 2 ACL.

```
device(config)# mac access-list extended macl
```

3. Configure the Layer 2 ACL for mirroring.

```
device(conf-macl-ext)# seq 10 permit host 0010.9400.0010 host 0010.9400.0014 vlan 20
count mirror
device(conf-macl-ext)# seq 20 permit host 0010.9400.0011 host 0010.9400.0015 vlan 20
count mirror
device(conf-macl-ext)# seq 30 permit host 0010.9400.0012 host 609c.9f01.58cb vlan 21
count mirror
device(conf-macl-ext)# seq 40 permit host 0010.9400.0013 host 609c.9f01.58cb vlan 22
count mirror
device(conf-macl-ext)# seq 50 permit any any count mirror
```

4. Return to global configuration mode.

```
device(conf-macl-ext)# exit
```

5. Create an extended IPv4 ACL.

```
device(config)# ip access-list extended ipv4acl
```

6. Configure the IPv4 ACL for mirroring.

```
device(conf-ipv4acl-ext)# seq 10 permit ip host 11.12.13.14 any count mirror
```

7. Return to global configuration mode.

```
device(conf-ipv4acl-ext)# exit
```

8. Associate the ACL destination mirror port.

```
device(config)# acl-mirror source ethernet 0/1 destination port-channel 1
```

9. Enter configuration mode for the source mirror port.

```
device(config)# interface ethernet 0/4
```

10. Bind the Layer 3 IP ACL to the source mirror port.

- a. Bind the Layer 2 ACL to the source mirror port.

```
device(conf-if-eth-0/1)# mac access-group macl in
```

- b. Bind the IPv4 ACL to the source mirror port.

```
device(conf-if-eth-0/1)# ip access-group ipv4acl in
```

11. Return to privileged exec mode.

```
device(conf-if-eth-0/1)# end
```

12. Verify the configuration.

```
device# show statistics access-list interface ethernet 0/1 in
mac access-list macl on Ethernet 0/1 at Ingress (From User)
  seq 10 permit host 0010.9400.0010 host 0010.9400.0014 vlan 20 count mirror
(105555094236 frames)
  seq 20 permit host 0010.9400.0011 host 0010.9400.0015 vlan 20 count mirror
(105555103123 frames)
  seq 30 permit host 0010.9400.0012 host 609c.9f01.58cb vlan 21 count mirror
(105555072247 frames)
  seq 40 permit host 0010.9400.0013 host 609c.9f01.58cb vlan 22 count mirror
(105555083432 frames)
  seq 50 permit any any count mirror (0 frames)
```

13. Save the configuration.

```
device# copy running-config startup-config
```

Inbound ACL-based mirroring to the insight interface configuration example (Layer 2)

```
device# configure terminal
device(config)# mac access-list extended macl
```

```

device(conf-macl-ext)# seq 10 permit host 0010.9400.0010 host 0010.9400.0014 vlan 20
count mirror
device(conf-macl-ext)# seq 20 permit host 0010.9400.0011 host 0010.9400.0015 vlan 20
count mirror
device(conf-macl-ext)# seq 30 permit host 0010.9400.0012 host 609c.9f01.58cb vlan 21
count mirror
device(conf-macl-ext)# seq 40 permit host 0010.9400.0013 host 609c.9f01.58cb vlan 22
count mirror
device(conf-macl-ext)# seq 50 permit any any count mirror
device(conf-macl-ext)# exit
device(config)# acl-mirror source ethernet 0/1 destination port-channel 1
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# mac access-group macl in
device(conf-if-eth-0/1)# end
device# show statistics access-list interface ethernet 0/1 in
device# copy running-config startup-config

```



Note

Only the Layer 2 ACL creation is shown in this example.

Insight interface traffic management and QoS

An Insight-enabled port channel may be the destination endpoint of multiple mirroring rules or forwarded traffic, resulting in a very high rate of traffic. The maximum Insight Interface bandwidth on all platforms is 10Gbps, and although the TVPM runs on two vCPUs, the cumulative traffic may result in the fast plane ASIC dropping some egress traffic at the Insight Interface.

From a traffic management perspective, QoS for an Insight Interface is similar to QoS for a regular port. If required, SLX conventional egress traffic rate-limiting or typical QoS features may be applied to the Port Channel with an enabled Insight Interface. The difference is the QoS configuration is applied to an Insight Interface LAG (port-channel).

Consider the following when you configure this feature:

- The TM supports egress scheduling, rate shaping, WRED, and ingress buffer management for insight interface.
- TM egress scheduling and shaping for the insight interface must be configured under a port-channel interface.
- The QoS configuration is automatically applied to all ports in the port-channel.
- Follow the same configuration procedures for ingress buffer management and WRED as you would with a standard port.

For more information, see [Configuring QoS egress scheduling](#) on page 93.

QoS egress scheduling

QoS egress scheduling works under the following rules, restrictions, and limitations:

- You must use a credit request/grant mechanism to perform egress scheduling QoS.
- The maximum credit size is 1024 Bytes.
- For each egress port there are 8 Virtual Output Queues (VOQs) allocated on each ingress transmit module (TM) core to support 8 priorities.
- Egress scheduling supports strict priority (SP), weighted fair queue (WFQ), and mixed mode scheduling.

- You can specify weighted for each VOQ only in WFQ mode.
- Fair queue (FQ) scheduling between VOQs from different TMs and with the same priority is permitted.

The figure below illustrates a QoS egress scheduling scheme.

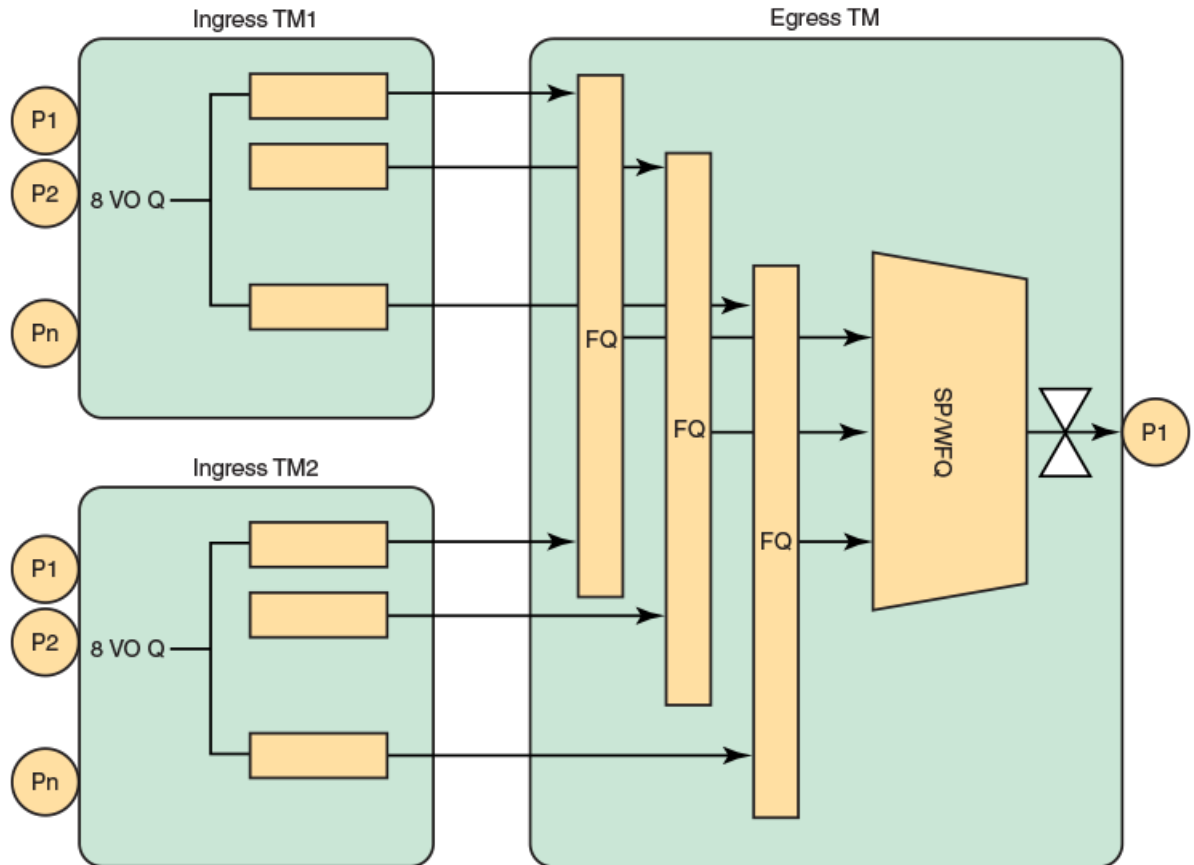


Figure 9: QoS egress scheduling scheme

In the figure above P1 is the insight interface.

See the topic [Configuring QoS egress scheduling](#) on page 93 for configuration information.

QoS rate shaping

QoS rate shaping allows you to limit egress traffic to a specified rate.

Rate shaping works under the following rules and limitations:

- If there is a higher data rate than the configured shaping rate, traffic is kept at the ingress VOQ.
- The ingress TM tail drops packets if the queue is full.
- Accuracy is +/- 3%.
- Configuration granularity is 1KB.

Configuring QoS egress scheduling

Follow the below steps to configure QoS egress scheduling.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode for port-channel 1.

```
device(config)# interface port-channel 1
```

3. Specify the option for strict priority mode to determine strict priority queues.

```
device(port-channel-1)# qos queue scheduler strict-priority 4
```

There are seven traffic classes. Specify the weight for the priority If the priority is in WFQ mode.

4. Return to privileged exec mode.

```
device(port-channel-1)# end
```

5. Verify the configuration.

```
device# show qos interface port-channel 1
```

6. View the VOQ statistics.

```
device# show tm voq-stat ingress-device ethernet 0/15 egress-port ethernet 0/125
```

```
VOQ-Counters:
```

```
=====
```

```
Priority 0
```

```
-----
EnQue Pkt Count          0
EnQue Bytes Count        0
Total Discard Pkt Count  0
Total Discard Bytes Count 0
Current Queue Depth      0
Maximum Queue Depth since Last read 0
```

```
Priority 1
```

```
-----
EnQue Pkt Count          0
EnQue Bytes Count        0
Total Discard Pkt Count  0
Total Discard Bytes Count 0
Current Queue Depth      0
Maximum Queue Depth since Last read 0
```

```
Priority 2
```

```
-----
EnQue Pkt Count          0
EnQue Bytes Count        0
Total Discard Pkt Count  0
Total Discard Bytes Count 0
Current Queue Depth      0
Maximum Queue Depth since Last read 0
```

```
Priority 3
```

```
-----
EnQue Pkt Count          0
```

```

EnQue Bytes Count          0
Total Discard Pkt Count    0
Total Discard Bytes Count  0
Current Queue Depth        0
Maximum Queue Depth since Last read  0

Priority 4
-----
EnQue Pkt Count            0
EnQue Bytes Count          0
Total Discard Pkt Count    0
Total Discard Bytes Count  0
Current Queue Depth        0
Maximum Queue Depth since Last read  0

Priority 5
-----
EnQue Pkt Count            0
EnQue Bytes Count          0
Total Discard Pkt Count    0
Total Discard Bytes Count  0
Current Queue Depth        0
Maximum Queue Depth since Last read  0

Priority 6
-----
EnQue Pkt Count            0
EnQue Bytes Count          0
Total Discard Pkt Count    0
Total Discard Bytes Count  0
Current Queue Depth        0
Maximum Queue Depth since Last read  0

Priority 7
-----
EnQue Pkt Count            0
EnQue Bytes Count          0
Total Discard Pkt Count    0
Total Discard Bytes Count  0
Current Queue Depth        0
Maximum Queue Depth since Last read  0

```

7. Save the configuration.

```
device# copu running-config startup-config
```

QoS egress scheduling configuration example

```

device# configure terminal
device(config)# interface port-channel 1
device(port-channel-1)# qos queue scheduler strict-priority 4
device(port-channel-1)# end
device# show qos interface port-channel 1
device# show tm voq-stat ingress-device ethernet 0/15 egress-port ethernet 0/125
device# copy running-config startup-config

```

Troubleshooting port-mirroring

Follow these high level steps to troubleshoot port-mirroring.

1. MAC counters on source and destination interfaces can be verified by running the command : **show interface ethernet slot/port**.
2. To see if packets are sent to a destination queue, VOQ counters can be verified by running the command: **show tm voq-stat**.
3. Management commands include:
 - a. **show interface port-channel <ID>**
 - b. **show port-channel <ID>**
 - c. **show interface stats [brief | detail]**
 - d. **show interface ethernet 0/125**
 - e. **show interface ethernet 0/126**

Troubleshooting port-mirroring

Use these example in debugging port mirroring.

Configuring port mirroring from interface 0/2 to 0/1.

```
evlce(config)# monitor session 1
device(config-session-1)# source ethernet 0/2 destination ethernet 0/1 direction rx
```

Display the MAC counters on the ingress interface:

```
device# show interface ethernet 0/2
Ethernet 0/2 is up, line protocol is up (connected)
Receive Statistics:
  1000 packets, 128000 bytes
  Unicasts: 1000, Multicasts: 0, Broadcasts: 0
...
(Output is truncated)
```

Display the MAC counters on the mirrored interface:

```
device# show interface ethernet 0/1 (Output is trimmed for brevity)
Ethernet 0/1 is up, line protocol is up (connected)
Transmit Statistics:
  1000 packets, 128000 bytes
  Unicasts: 1000, Multicasts: 0, Broadcasts: 0
  Underruns: 0
...
(Output is truncated)
```

SLX-OS VM commands

Use these commands on the SLX platform to help troubleshoot the VM.

Display general interface information

```
device# show interface stats brief
```

Interface	Packets		Error		Discards		CRC	
	rx	tx	rx	tx	rx	tx	rx	

=====	=====	=====	=====	=====	=====	=====	=====
Po 1	16	2	0	0	0	0	0
Eth 0/1	0	0	0	0	0	0	0
Eth 0/125	8	3	0	0	0	0	0

Display port-channel information

```
device# show interface port-channel 1
Port-channel 1 is up, line protocol is up
Hardware is AGGREGATE, address is 748e.f88f.5ffd
  Current address is 748e.f88f.5ffd
Description: Insight port-channel
Interface index (ifindex) is 671088641
Minimum number of links to bring Port-channel up is 1
MTU 2500 bytes
LineSpeed Actual      : 10000 Mbit
Allowed Member Speed : 10000 Mbit
Priority Tag disable
IPv6 RA Guard disable
Last clearing of show interface counters: 03:43:52
Queueing strategy: fifo
Receive Statistics:
  0 packets, 0 bytes
  Unicasts: 0, Multicasts: 0, Broadcasts: 0
  64-byte pkts: 0, Over 64-byte pkts: 0, Over 127-byte pkts: 0
  Over 255-byte pkts: 0, Over 511-byte pkts: 0, Over 1023-byte pkts: 0
  Over 1518-byte pkts(Jumbo): 0
  Runts: 0, Jabbers: 0, CRC: 0, Overruns: 0
  Errors: 0, Discards: 0
Transmit Statistics:
  0 packets, 0 bytes
  Unicasts: 0, Multicasts: 0, Broadcasts: 0
  Underruns: 0
  Errors: 0, Discards: 0
Rate info:
  Input 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
  Output 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
Time since last interface status change: 03:43:52
```

```
device# show running-config interface port-channel 1
interface Port-channel 1
 no vlag ignore-split
 description Insight port on MM1
 no shutdown
```

```
device# show port-channel
Static Aggregator: Po 1
Aggregator type: Standard
Eth 0/125
```

Optional keywords are **summary**, **detail**, and **load-balance**.

Display Ethernet interface information

```
device# show interface ethernet 0/125
Ethernet 0/125 is up, line protocol is down (link protocol down)
Hardware is Ethernet, address is 0027.f817.12fe
  Current address is 0027.f817.12fe
```



```
Pluggable media not present
Interface index (ifindex) is 4704206921
MTU 2500 bytes
LineSpeed Actual      : Nil
LineSpeed Configured : Auto, Duplex: Full
Priority Tag disable
IPv6 RA Guard disable
Last clearing of show interface counters: 03:46:22
Queueing strategy: fifo
Receive Statistics:
  0 packets, 0 bytes
  Unicasts: 0, Multicasts: 0, Broadcasts: 0
  64-byte pkts: 0, Over 64-byte pkts: 0, Over 127-byte pkts: 0
  Over 255-byte pkts: 0, Over 511-byte pkts: 0, Over 1023-byte pkts: 0
  Over 1518-byte pkts(Jumbo): 0
  Runts: 0, Jabbers: 0, CRC: 0, Overruns: 0
  Errors: 0, Discards: 0
Transmit Statistics:
  0 packets, 0 bytes
  Unicasts: 0, Multicasts: 0, Broadcasts: 0
  Underruns: 0
  Errors: 0, Discards: 0
Rate info:
  Input 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
  Output 0.000000 Mbits/sec, 0 packets/sec, 0.00% of line-rate
Time since last interface status change: 03:46:22
```

TPVM

TPVM enables users to run applications such as Docker Container, syslog server, SNMP server, and RESTful applications, among others. TPVM runs as a separate, independent virtual machine (VM), sharing the host CPU, RAM, hard disk drive, and management resources with SLX-OS.

Extreme devices are shipped with the TPVM firmware, but it is not installed by default.

Supported third-party applications, packages, and hardware

Note the following basic support and limitations for TPVM.

Third-party applications

- Packet capture applications
- RESTful support to access SLX-OS
- perfSONAR
- ARPsponge

Third-party packages

- Packages installed by default on the TPVM
- RESTful application: Chrome browser – GUI RESTful access
- RESTful application: cURL – Command line RESTful access
- Tcpcap: Command line packet-capture utility
- Tshark: Command line packet-capture utility

- Wireshark: GUI packet-capture utility
- Datadog or Splunk – External analytics software service

**Note**

Extreme SLX-OS provides support for built-in applications (third-party packages shipped with the SLX-OS) that are listed in the *Extreme SLX-OS Management Configuration Guide*.

- Extreme is committed to providing limited support for the interoperability of these applications with Extreme application interfaces.
- Extreme does not provide support for the application configuration, functionality, or deficiencies.
- Extreme does not provide any support for applications not listed in the *Extreme SLX-OS Management Configuration Guide*.

Hardware

The TPVM has 4 GB of RAM and 240GB of solid-state disk (SSD) memory, which limits the amount of data captured through packet capture applications. To overcome this limitation, Extreme provides support for the Network File System (NFS) mount of an external drive.

perfSONAR

perfSONAR (Performance focused Service Oriented Network monitoring ARchitecture) is an open-source, active network measurement toolkit that provides federated coverage of paths and helps establish end-to-end user expectations.

To identify network problems, it is important to compare active measurements against predefined notions of successful networks. Performance probes are placed in paths of interest, such as campus network endpoints, demarcations between networks, within carrier points of presence, at exchange points, and near data resources such as storage and computing elements.

To provide measurement baselines, some 2000 perfSONAR instances are deployed worldwide, representing around 300 domains, and many of which are available for the open testing of key measures of network performance. This global infrastructure helps to identify and isolate problems as they occur, making the role of supporting network users easier for engineering teams, and increasing productivity in the use of network resources.

perfSONAR provides a uniform interface that allows for the scheduling of measurements, storage of data in uniform formats, and scalable methods to retrieve data and generate visualizations. This extensible system can be modified to support new metrics, with a variety of ways to present data.

perfSONAR features

perfSONAR supports the following features and tools:

- Active measurement with scheduled tests
- Tools: Bwctl (lperf, iperf3, Nuttcp tests), ping, OWAMP tests (one-way latency), traceroute, tracepath
- Visualizations of stored data (MadDash)
- Directory service (to find perfSONAR instance around the world)

TPVM deployment considerations

Although SLX-OS allows perfSONAR to run on TPVM, it is recommended that this application be run on a dedicated server to mitigate risks posed by the VM environment, for the following reasons:

1. *Time keeping:* Some virtualization environments implement clock management as a function of the hypervisor and VM communication channel, rather than using a stabilization daemon such as NTP. This could result in timing skipping forward or backward, making it generally unpredictable for measurement.
2. *Data path:* Additional hypervisor layers can cause undesired latency.
3. *Resource management:* Because VMs share physical hardware and might get swapped, this might introduce additional errors in network performance measurements.

Reason (2) is mitigated in TPVM deployments by directly assigning the insight interface to the VM.

Reason (3) can be potentially mitigated by pinning one or more CPU cores to the VM.

Reason (1) can also be mitigated, such as by running NTP between guest and host, but this still not provide sufficient accuracy.

The perSONAR development team has identified several use cases that can work in VM environments, provided the known issues are mitigated. However, the high-speed throughput and OWAMP tests do not perform well.

ARP sponge

ARP sponge is an application that snoops on ARP packets on the Virtual Private LAN Service (VPLS) domain.

ARP sponge listens for ARP traffic. When the number of ARP requests for a certain IP address exceeds a threshold, ARP sponge sends out an ARP reply for that address that uses its own MAC address. This achieved by using the insight LAG to the bridge domain as an AC endpoint. All ARP traffic received by the VPLS instance is flooded to the insight LAG as well.

TPVM Installation and Management

TPVM installation and a variety of management details are described.

Installation Overview

The TPVM package is available separately from the SLX-OS software, and may be downloaded from the SLX-OS Release Server. This decoupling enables faster turnaround on enhancements and bug fixes, while reducing the file size of the SLX-OS distribution.

Verify that the TPVM firmware package is available in the SLX-OS filesystem's `/tftpboot/SWBD2900` directory; download if necessary. The TPVM firmware installation may then be run.

The TPVM firmware package is not updated as a part of a subsequent SLX-OS release firmware download. The TPVM firmware package installation procedure is independent of the SLX-OS release upgrade procedure. Manually copy the TPVM firmware package compatible with the SLX-OS release, uninstall the existing TPVM firmware, and install the new TPVM firmware package.

If you have already installed a TPVM version and plan to upgrade to the next SLX-OS release version, it is recommended that you first uninstall the current TPVM firmware. Once a subsequent SLX-OS release firmware download is completed, install the TPVM firmware package that is compatible with that SLX-OS release. This option avoids TPVM firmware incompatibility with subsequent SLX-OS release firmware. In case of an upgrade to the next SLX-OS release, the current TPVM firmware remains unchanged.

Execute the **tpvm install** command to install the package.

**Important**

The installation is disruptive, and any data saved on the TPVM partition is erased. You must save any data manually before executing the **tpvm install** command.

You may also use the `tpvm-deploy` command to install and configure TPVM and the Insight Interface. See [Using tpvm deploy](#) on page 108 for additional information about this container command.

After the installation, you can start and stop the image by means of the **tpvm start** and **tpvm stop** commands, respectively. To start the TPVM image automatically in subsequent reboots, use the **tpvm auto-boot enable** command. (TPVM may not come up if there are any issues with booting SLX-OS.)

Once the TPVM image is running, you can download user-specific applications by copying them to the TPVM partition and starting them manually.

To uninstall the TPVM image and release its resources, use the **tpvm uninstall** command.

**Important**

When TPVM is re-installed, any user applications are deleted.

Resources and default XML configuration

TPVM runs the Linux 4.15 64-bit kernel, and has 4 GB of RAM. On the SLX 9540 and 9640, the second SSD provides an additional dedicated 120 GB of memory, and shares one of the 8 CPU cores with the SLX-OS. The 9150 SSD allocates an additional 64 GB of RAM to the TPVM, and shares one of 8 CPU cores with SLX-OS. XML is used as the file format for storing the total configuration, including domain, network, storage, and other elements. The storage file is used by QEMU/Libvirt to instantiate TPVM, and cannot be edited by the user.

Resource usage

From the Linux host's perspective, TPVM appears as a process. All commands to check TPVM resources and control TPVM are executed from the host and have administrative (root) restrictions.

To check TPVM resource utilization, use the following commands:

- **ps aux | grep TPVM**
- **top -p pid**
- **cat /proc/ pid /***

Console access

A console daemon runs on the host and opens a console connection to TPVM on the 9540. You can switch the console connection between host, SLX-OS, and TPVM using the following key sequences, respectively.

- Host: **Ctrl + y + 1**
- SLX-OS: **Ctrl + y + 2**
- TPVM: **Ctrl + y + 3**

For information on accessing the console on the 9150 and other baremetal platforms, see [TPVM on the SLX 9150 series](#) on page 117.



Note

By default, the console is connected to SLX-OS.

TPVM can be accessed through the eth0 (management) interface. The eth0 interface connects to the outside network through the host physical interface, which makes it appear as a normal host to the rest of the network. SSH or Telnet access to TPVM is provided through the IP address of the eth0 interface configured on TPVM.

IP address management

The assignment of a TPVM IP address to a management interface uses DHCP by default. However, the user can also assign a static address and a default gateway to the TPVM eth0 interface by using the **ifconfig** command. See [Assigning a static IP address on the TPVM Linux OS](#) on page 116.

Communication between TPVM and SLX-OS

An HTTP RESTful interface is provided for accessing the running configuration, interface statistics, interface states, and all system-related information. TPVM is prepackaged with a RESTful client to support, for example, cURL (command line RESTful access utility), to extract information from SLX-OS. cURL uses HTTP methods (such as GET, PUT) to extract and modify configuration information so long as the requesting user is authenticated correctly.

The following example shows a simple request format from TPVM:

```
curl -s -u admin:password http://ip-addr/rest/config/running/ . . .
```

In addition to cURL, Advanced Rest Client, a Chrome-based RESTful client application, is prepackaged inside TPVM and is accessed through a browser interface.

Both HTTP and HTTPS secure access are enabled.

NFS mount support

TPVM supports the NFS mount of an external drive to support the storage of captured data.

Packages support and applications

No configuration is needed on TPVM to support RESTful access with cURL and Google-chrome.

In addition, TPVM comes with Tcpdump, Tshark, and Wireshark prepackaged to support packet capture.

Users or administrators can use the **apt-get** command with options to upgrade, update, purge, or remove (to downgrade to an older version). In addition, applications can be downloaded to provide a development environment that allows users or administrators to build their own applications, development tools (gdb, glibc (e.g. ANSI-C and POSIX), and gcc for C/C++ . Similarly, python development tools can also be downloaded.

Containers

The following container binaries have been tested with TPVM:

- Docker container: docker-1.13.0
- Linux container: LXC 1.0

The above binaries do not come prepacked with TPVM. Use the **wget** or **apt-get** commands to install, upgrade, or downgrade Docker and Linux container binaries or packages in TPVM.

Frequently used commands

Command	Description
auto-boot	Enables or disables start of TPVM at next boot.
disk	Supports TPVM disk operations.
install	Installs TPVM.
password	Updates the root password.
start	Starts TPVM.
stop	Stops TPVM.
uninstall	Uninstalls TPVM.

Installing TPVM

The TPVM package is available separately from the SLX-OS software, and may be downloaded from the SLX-OS Release Server. This decoupling enables faster turnaround on enhancements and bug fixes, while reducing the file size of the SLX-OS distribution. (Prior to SLX-OS 18r.2.00, the TPVM package was released in conjunction with the SLX-OS packages.)



Important

The installation is disruptive, and any data saved on the TPVM partition is erased. You must save any data manually before executing the **tpvm install** command.

Please also note:

- The TPVM package must be downloaded to the device prior to starting the installation.
- The TPVM package is generated at the following path:

```
<slxos release dir>/SWBD2900/tpvm-3.0.0-0.amd64.deb
```

The following table lists the TPVM package names and locations for relevant SLX-OS releases.

Table 17: TPVM package names and locations for relevant SLX-OS releases

SLX-OS release	TPVM package name	TPVM package location
slxos16r.x.xx	vm-swbd2900-1.0.0-1.i386.deb	<slxos release dir>/ SWBD2900/vm- swbd2900-1.0.0-1.i386.d eb
slxos17r.1.xx	vm-swbd2900-1.0.0-1.i386.deb	<slxos release dir>/ SWBD2900/vm- swbd2900-1.0.0-1.i386.d eb
slxos17r.2.xx	vm- swbd2900-1.0.0-1.amd64.deb	<slxos release dir>/ SWBD2900/vm- swbd2900-1.0.0-1.amd64. deb
slxos18r.1.xx	vm- swbd2900-1.0.0-1.amd64.deb	<slxos release dir>/ SWBD2900/vm- swbd2900-1.0.0-1.amd64. deb
slxos18r.2.xx	tpvm-2.0.0-0.amd64.deb	<TPVM release url>/ tpvm2.0.0/ tpvm-2.0.0-0.amd64.deb (see NOTE)
slxos 20.1.xx	tpvm-3.0.0-0.amd64.deb	<TPVM release url>/ tpvm3.0.0/ tpvm-3.0.0-0.amd64.deb



Note

Removed from SLX-OS release folder and posted on release server for access through separate URL.

Installing and upgrading TPVM

This task performs a basic installation, removal, and re-installation of TPVM.

1. Switch to the Linux shell from the SLX-OS CLI.

```
device# start-shell
```

2. Enter the Linux admin user shell.

```
[admUser@SLX]# su  
password: <password>
```

The default password is "fibranne".

3. Remove the existing TPVM package at the following path in the device's SLX-OS VM, at the Linux shell login prompt.

```
[admUser@SLX]# rm -rf /tftboot/SWBD2900/vm-swbd2900-*.deb
```

4. Using SCP or FTP, copy the new TPVM package that is compatible with the SLX-OS release from the release server, and copy it to the SLX-OS VM /tftboot/SWBD2900/ folder.

- Return to the SLX-OS CLI shell.

```
root# exit
admin# exit
```

- Stop TPVM if it is running.

```
device# tpvm stop
```

- Uninstall existing TPVM firmware.

```
device# tpvm uninstall
```

- Re-install appropriate TPVM firmware.

```
device# tpvm install
```

Once the new TPVM image is running, you can download user-specific applications by copying them to the TPVM partition and starting them manually.

Upgrading the TPVM package from 18r.1.xx or 18r.2.xx to 20.1.1

Use the following procedure to upgrade the TPVM package from 18r.1.xx or 18r.2.xx to 20.1.1.

- Uninstall the existing TPVM package.

```
device# tpvm uninstall
```

- Upgrade the device with the current release by using the **firmware download** command, with options as applicable.

- Remove the existing TPVM package at the following path in the device's SLX-OS VM, at the Linux shell login prompt.

```
[admUser@SLX]# rm -rf /tftpboot/SWBD2900/vm-swbd2900-*.deb
```

- Using SCP or FTP, copy the following package from the release/build server (<TPVM release URL>/tpvm3.0.0/tpvm-3.0.0-0.amd64.deb) to the following directory on the device: /tftpboot/SWBD2900/.

- Install the new TPVM package.

```
device# tpvm install
```

- Confirm the installation status and start TPVM.

```
device# show tpvm status
device# tpvm start
```

Downgrading the TPVM package from 20.1.1 to 18r.2.xx or 18r.1.xx_SLXR

Use the following procedure to downgrade the TPVM package from 20.1.1 to 18r.2.xx or 18r.1.xx.

- Uninstall the existing TPVM package.

```
device# tpvm uninstall
```

- Downgrade the device to the 18r.2.xx or 18r.1.xx release by using the **firmware download** command, with options as applicable.

- Remove the existing TPVM package at the following path in the device's SLX-OS VM, at the Linux shell login prompt.

```
[admUser@SLX]# rm -rf /tftpboot/SWBD2900/tpvm-*.deb
```

- Using SCP or FTP, copy the following package from the release/build server (<SLX-OS release URL>/slxos18r.1.xx>/SWBD2900/vm-swbd2900-1.0.0-1.amd64.deb or slxos18r.2.xx>/SWBD2900/vm-swbd2900-2.0.0-0.amd64.deb) to the following directory on the device: /tftpboot/SWBD2900/.

5. Install the new TPVM package.

```
device# tpvm install
```

6. Confirm the installation status and start TPVM.

```
device# show tpvm status
device# tpvm start
```

Using the *tpvm* command

The **tpvm** command is available at the SLX-OS CLI on a device.

The **tpvm** command, in privileged EXEC mode, allows you to manage TPVM with a variety of subcommands that do the following:

- Install, start, stop, and uninstall TPVM
- Specify the default behavior when SLX-OS boots
- Add or remove disks and show the disk information
- Print out IP addresses set on TPVM
- Change the root password on TPVM
- Use the help keyword for details on all options

Install TPVM:

```
tpvm install
device# tpvm install
```

Uninstall TPVM:

```
tpvm uninstall [ force ]
force: clear installation or uninstallation error(s) then try to uninstall (forcefully)
device# tpvm uninstall
uninstallation succeeds
```

To force the clearing of installation or uninstallation errors, use the **force** keyword:

```
device# tpvm uninstall
TPVM uninstallation failed
device# tpvm uninstall force
uninstallation succeeds
```

To start TPVM:

```
tpvm start
device# tpvm start
start succeeds
```

To stop TPVM:

```
tpvm stop
device# tpvm stop
stop succeeds
```

To automatically start TPVM at the next reboot of SLX-OS use **auto-boot enable**:

```
tpvm auto-boot enable

device# tpvm auto-boot enable
```

To prevent TPVM from starting at the next reboot of SLX-OS:

```
tpvm auto-boot disable

device# tpvm auto-boot disable
auto-boot disable succeeds
```



Note

In this case, the **tpvm start** command is required to enable TPVM.

To display the current status of TPVM, or any errors, use the following:

```
show tpvm status [ clear-tag <tag name> ]

clear-tag: clear the runtime error when the 'command' ran

device# show tpvm status
TPVM is running, and AutoStart is disabled on this host.
```

To clear errors use the **clear-tag**<tag name> keywords, where the error in this example is "vm_disks":

```
device# tpvm start
start succeeds

device# show tpvm status
TPVM had runtime error(s) -- these error(s) seem not fatal, and the operation(s) could be
retryable
vm_disks: virsh list timed out. TPVM cannot transit to 'running' state

TPVM is installed but not running, and AutoStart is disabled on this host.

device# show tpvm status clear-tag vm_disks
TPVM is installed but not running, and AutoStart is disabled on this host.
```



Note

The runtime error can be also removed automatically when the same subcommand succeeds.

To add a new disk to TPVM, use the following commands:

```
tpvm disk add name <vd[b-x] | auto> size <number | number[bkmg]>
```

name: The disk name added to TPVM.

The name must be 'vd[b-x]' or 'auto'.

Note. The disk name must be the next disk if it's not 'auto'. For example, if the last disk added to the system is 'vdb', the disk name must be 'vdc'. When 'auto' is used, the system automatically assigns the next disk name.

size: Any positive number.

Also the following suffix can be added to the end.

```
b or B for bytes
k or K for KiB bytes
m or M for MiB bytes
```

```

        g or G for GiB bytes
    Note. When no suffix is used, the size is taken as GiB bytes. For example, '5'
    means '5g'.
device# tpvm disk add name auto size 10g
disk add succeeds

device# tpvm disk add name vdd size 512m
disk add succeeds

```



Note

The maximum number of disks is currently 3. If the number of allocated disks exceeds this list, the **add_disk** keyword fails. Also, the total disk capacity is limited to 50 Gbytes on the SLX 9540. If you exceed this limit when you create a disk, the **add_disk** keyword fails.

Use the **disk remove** command to remove an additional disk from TPVM:

```

tpvm disk remove name <vd[b-x] | auto>

name: The disk name removed from TPVM.
    The name must be 'vd[b-x]' or 'auto'
    Note. When 'auto' is passed, the system removes the latest disk automatically.
    Otherwise, the disk name must be the last disk added to the system.
    For example, if the last disk added to the system is 'vdx', the
    disk removed from the system must be this disk, 'vdx'.
device# tpvm disk remove name auto
'umount' is needed before this disk is removed. Continue? [y/n]: y
disk remove succeeds

device# tpvm disk remove name vdc
'umount' is needed before this disk is removed. Continue? [y/n]: y
disk remove succeeds

```



Note

Disks must be unmounted before removal from the system. Otherwise, the next added disk will be labeled incorrectly. If the system falls, TPVM must be rebooted to recover.

Display disk information:

```

show tpvm disk name <vd[b-x] | all>

name: The disk name whose information is shown.
    The name must be 'vd[b-x]' or 'all'
    Note. When 'all' is passed, the information about all disks is shown.
device# show tpvm disk
Value for 'name' : all
disk: vdb
Capacity: 10.00 GiB
Allocation: 196.00 KiB

total:
Capacity: 100.00 GiB
Allocation: 10.00 GiB
Available: 90.00 GiB

device# show tpvm disk name vdb
disk: vdb
Capacity: 10.00 GiB
Allocation: 196.00 KiB

```

```
total:
Capacity: 100.00 GiB
Allocation: 10.00 GiB
Available: 90.00 GiB
```

Display IPv4 and IPv6 addresses:

```
show tpvm ip-address

device# show tpvm ip-address
IPv4:
eth0 10.24.7.149
IPv6:
eth0 fe80::629c:9fff:fe01:fe43
eth1 fe80::7:d0ff:fe02:100
```



Note

The **show_ip_addr** parameter requires the *qemu-guest-agent* package on TPVM. If this package is removed, the operation fails.

Change the root password on TPVM:

```
tpvm password

device# tpvm password
root password: ****
re-enter root password: ****
password succeeds
```

Using *tpvm deploy*

The `tpvm deploy` command is a container command, rolling several SLX commands into one, to perform TPVM and Insight Interface set up and configuration.

Command Overview

The `tpvm deploy` command performs the following installation and configuration operations:

- Installation of TPVM
- TPVM Networking set up
- Enable Passwordless ssh to TPVM from `root@slx`
- Enable passwordless “sudo” inside TPVM
- Set the TPVM password for the default “admin” user
- Set TPVM autoboot
- Start or boot the TPVM

There are two important pre-requisites to use the `tpvm deploy` command:

- TPVM Debian package image – available in the `/tftpboot/SWBD2900` folder. If TPVM has already been installed, then you may skip this step.
- An Advanced Features License. Use the following command to activate the license: `license eula accept ADVANCED_FEATURES`.

TPVM Installation

Verify the presence of the TPVM firmware package in the `SLXVM /tftpbboot/SWBD2900` directory. If the latest version is not there, download before running the `TPVM deploy` command.

The `tpvm deploy` command begins with the standard TPVM installation.

TPVM Networking Setup

By default the TPVM management interface `eth0` is configured to acquire an IP Address via DHCP, whereas the `eth1` address is manually configured by adding a static entry in `/etc/network/interfaces`.

One of these interfaces may be configured using `tpvm deploy`. The `tpvm deploy` command uses the interface and ip address parameters to configure the TPVM interface. The interface not configured using `tpvm deploy` is put into manual mode with no IP address assigned, regardless of the current or previous state.

Passwordless SSH

The passwordless parameter within `tpvm deploy` allows you to configure ssh access from the root user account on the SLX-OS to TPVM without a password. For example:

```
root@SLX# ssh -o "StrictHostKeyChecking no" admin@10.23.30.153
```

When using the passwordless parameter, please note the following:

- Passwordless ssh capability will be retained across firmware downgrade and upgrade.
- Passwordless ssh capability is lost in the case of a netinstall where `tpvm deploy` is used, regardless of whether TPVM is reinstalled or retained from previous install.
- The SLX-OS must be running and a compatible version of TPVM currently installed.

Passwordless SUDO

The TPVM default user is `admin` with `sudo` privileges. The `tpvm deploy` command configures TPVM so `sudo` for this user does not ask for a password. Setting this parameter once will persist for the lifetime of the TPVM.

If not set, the default behavior requires a password for `sudo` activities, as dictated by the Ubuntu 16.04 LTS Server Operating System.

TPVM Password

TPVM ships with `admin/password` as the default login credential. To automate the TPVM setup and achieve one touch provisioning of TPVM, this optional parameter will set the password for the TPVM `admin` user account. Setting this parameter once will persist for the lifetime of the TPVM.

TPVM Auto-boot

This option will restart the TPVM image automatically in subsequent reboots, such as an SLX-OS start on a Baremetal platform, or a HOST start on a VM-based platform.

TPVM Start

After configuring the TPVM as described above, `tpvm deploy` will start the TPVM. On a baremetal platform, a reboot of SLX will reboot TPVM. On a VM based platform, an SLX-OS reboot does not affect TPVM, however if the HOST reboots for any reason, TPVM also reboots.



Note

A firmware upgrade or downgrade will reboot the SLX-OS, but will not reboot TPVM.

Installing TPVM Using `tpvm deploy`

This task describes the use of the `tpvm deploy` command and parameters.

1. Switch to the Linux shell from the SLX-OS CLI.

```
device# start-shell
```

2. Enter the Linux admin user shell.

```
[admUser@SLX]# su
password: <password>
```

The default password is "fibranne".

3. Remove the existing TPVM package at the following path in the device's SLX-OS VM, at the Linux shell login prompt.

```
[admUser@SLX]# rm -rf /tftpboot/SWBD2900/vm-swbd2900-*.deb
```

4. Using SCP or FTP, copy the new TPVM package that is compatible with the SLX-OS release from the release server, and copy it to the SLX-OS VM `/tftpboot/SWBD2900/` folder.

5. Return to the SLX-OS CLI shell.

```
root# exit
admin# exit
```

6. Stop TPVM if it is running.

```
device# tpvm stop
```

7. Uninstall existing TPVM firmware.

```
device# tpvm uninstall
```

8. Use `tpvm deploy` to configure the network and management settings, and install TPVM.

```
device# tpvm deploy mgmt ipaddr 10.25.101.121/22 gw 10.25.100.1 admin-pwd mypassword
confirm mypassword
Starting TPVM deploy CLI, please DO NOT hit CTRL+C
Tpvm install started
..Tpvm is installed
Tpvm set_ip succeeds
Tpvm password succeeds
auto-boot enable succeeds
Tpvm is started
```

Above is one example of using TPVM deploy. The following are additional examples.

- Eth0 with DHCP and passwordless login

```
device# tpvm deploy interface mgmt ip-addr dhcp pwdless
```

- Eth0 with static IP and passwordless login.

```
device# tpvm deploy interface mgmt ip-addr 192.168.1.1/24 pwdless
```

- Eth0 with static IP, default gateway and passwordless login

```
# tpvm deploy interface mgmt ip-addr 192.168.1.1/24 gateway 192.168.1.100
pwdless
```

- Eth1 with static IP and passwordless login

```
device# tpvm deploy interface insight ip-addr 10.10.10.1/24 pwdless
```
- Eth1 with static IP, default gateway and passwordless login

```
device# tpvm deploy interface insight ip-addr 10.10.10.1/24 gateway 10.10.10.100
pwdless
```
- Eth1 with static IP, default gateway, passwordless login and new TPVM password setup

```
device# tpvm deploy interface insight ip-addr 10.10.10.1/24 gateway 10.10.10.100
pwdless password
admin123 admin123
```
- Eth0 with static IP, default gateway, passwordless login and new TPVM password setup

```
device# tpvm deploy interface mgmt ip-addr 10.10.10.1/24 gateway 10.10.10.100
password admin123
admin123
```
- Eth1 with Static IP, default gateway

```
device# tpvm deploy interface insight ip-addr 10.10.10.1/24 gateway
10.10.10.100
```

Docker containers

This section addresses the installation and management of Docker containers.

Installation

Complete the following steps to install the latest version of Docker under TPVM.

1. Install and export the missing Gnu Privacy Guard (GPG) key.

```
gpg --keyserver hkp://keyserver.ubuntu.com:80 --recv 1397BC53640DB551
gpg --export --armor 1397BC53640DB551 | apt-key add -
```

2. Add [arch=amd64] before `http://dl.google.com/linux/chrome/deb/ stable main` in the `/etc/apt/sources.list.d/google-chrome.list` file.
3. Update the repository: **apt-get -y update**
4. Install the CA certificates: **apt-get -y install apt-transport-https ca-certificates**
5. Add the new GPG key for Docker: **apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 --recv-keys 58118E89F3A912897C070ADBF76221572C52609D**
6. Create or update the `/etc/apt/sources.list.d/docker.list` file to contain the following string: `deb https://apt.dockerproject.org/repo ubuntu-trusty main`
7. Update the Advance Packaging Tool (APT) package index: **apt-get -y update**
8. Purge the old repository: **apt-get -y purge lxc-docker**
9. Verify that APT is pulling the Docker engine from the proper repository: **apt-cache policy docker-engine**
10. Install the Docker engine: **apt-get -y install docker-engine**
11. Start Docker service: **service docker start**
12. Verify the Docker installation by running the "hello-world" image: **docker run hello-world**

Docker Linux binaries can also be obtained from the following URL by means of the `wget` command:

- Docker script: <https://get.docker.com/>

After downloading the binaries, you extract the archive by using the **tar -xvzf docker-latest.tgz** command, which puts the binaries in a directory named `/docker` in the current location.

Depending upon the Docker engine version, you may have to set "execute" permission on the Docker daemon, by using the **chmod +x docker** command.

Docker requires the binaries to be installed in your host's `$PATH`. For example, you can move these binaries to `/usr/bin`.

Starting Docker

Start Docker by using the **service docker start &** command.

The docker daemon always runs as the root user, and binds to a UNIX socket instead of to a TCP port. By default, that UNIX socket is owned by the user "root", and therefore is accessible by means of the **sudo** or **root** commands.

If you (or your Docker installer) create a UNIX group called "docker" and add users to it, then the docker daemon makes the ownership of the UNIX socket read/writable by the docker group when the daemon starts. The docker daemon must always run as the root user, but if you run the docker client as a user in the docker group, then you do not need to add **sudo** to all the client commands.

Upgrading Docker

To upgrade your manual installation of Docker, first kill the docker daemon by using the **killall docker** command.

Running and monitoring Docker containers

You can start, stop, and monitor Docker containers by using the **docker** command. The following table lists frequently used commands.

Table 18: Frequently used docker commands

Command	Description
docker help	Lists supported Docker commands
docker --version	Displays the Docker version
docker create image	Creates a new container
docker run -i -t ubuntu /bin/bash	Instantiates a Docker container with bash shell and console connection
docker ps -a	List all Docker containers
docker attach container-id	Attach to a running Docker container
docker start container-id	Start/restart a particular Docker container
docker stop container-id	Stop a particular Docker container
docker rm \$(docker ps -a -q)	Delete all Docker containers
docker rmi \$(docker images -q)	Delete all Docker images

Linux containers

This section addresses the installation of Linux and creating and managing containers.

Installation

LXC 1.0 was tested with TPVM. The `lxc` package can be installed as root by means of the **`apt-get install lxc`** command.

Your system will then have all the LXC commands, all LXC templates, and also the python3 binding should you want to script LXC.

Creating containers

You can create privileged or unprivileged containers. (Only privileged containers were tested for this release.)

Privileged containers are containers created by root and running as root. They can be created as follows: **`sudo lxc-create -t download -n my-container`**

This creates a new privileged container "my-container" on TPVM, using an image based on the download template. The download template contains a list of distributions, versions, and architectures to choose from. Good example templates would be "ubuntu" and "trusty".

Running and monitoring containers

Once the container is created, start it by using the **`lxc-start -n my-container -d`** command.

You can then confirm its status by using either of the following commands:

- **`lxc-info -n my-container`**
- **`lxc-ls -f`**

You can access the `my-container` console by using the **`lxc-console -n my-container`** command.

You get a shell inside the container by using the **`lxc-attach -n my-container`** command.

Once done, you can stop the container by using the **`lxc-stop`** command, and remove it by using the **`lxc-destroy`** command:

- **`lxc-stop -n my-container`**
- **`lxc-destroy -n my-container`**

To confirm connectivity, attach to one of the containers and check network access by pinging a server accessible from the host:

- **`lxc-attach -n lxc1`**
- **`ping external-server`**

Utilities installation and management

cURL

cURL is a command-line RESTful access utility. The following table lists useful installation and management commands.

Table 19: cURL commands

Command	Description
<code>sudo apt-get update</code>	Updates any dependency packages
<code>sudo apt-get install curl</code>	Installs the latest cURL package, or specifies a previous version number
<code>sudo apt-get upgrade curl</code>	Upgrades to the latest cURL package

Google-chrome

Google-chrome is a graphical user interface RESTful access utility. The following table lists useful installation and management commands

Table 20: Google-chrome commands

Command	Description
<code>sudo apt-get update</code>	Updates any dependency packages
<code>sudo apt-get install google-chrome-stable</code>	Installs the latest Google-chrome package, or specifies a previous version number
<code>sudo apt-get upgrade google-chrome-stable</code>	Upgrades to the latest Google-chrome package

ifconfig and route

The **Ifconfig** and **route** utility commands are available by default in the Ubuntu/Debian package, and can be used without any additional package installation. The following table lists useful command options.

Table 21: ifconfig and route command options

Command	Description
<code>ifconfig eth0 Net-IP-Addr netmask <Net-IP-Addr-Mask></code>	Checks interface status and statistics
<code>route add -net Net-IP-Addr netmask Net-IP-Addr-Mask gw Gw-IP</code>	Adds a route
<code>route add default gw GW-IP</code>	Adds a default gateway

Ethtool

The Ethtool utility is used to get device information. The following table lists useful command options.

Table 22: Ethtool commands

Command	Description
<code>sudo apt-get update</code>	Updates any dependency packages
<code>sudo apt-get install ethtool</code>	Installs the latest Ethtool package, or specifies a previous version number
<code>sudo apt-get upgrade ethtool</code>	Upgrades to the latest Ethtool package

Tcpdump

Tcpdump is a command line utility that is used for packet capture by means of libpcap. The following table lists useful command options.

Table 23: Tcpdump commands

Command	Description
<code>sudo apt-get update</code>	Updates any dependency packages
<code>sudo apt-get install tcpdump</code>	Installs the latest Tcpdump package, or specifies a previous version number
<code>sudo apt-get upgrade tcpdump</code>	Upgrades to the latest Tcpdump package

Tshark

Tshark is a command line utility from the Wireshark community that is used for packet capture by means of libpcap. The following table lists useful command options.

Table 24: Tshark commands

Command	Description
<code>sudo apt-get update</code>	Updates any dependency packages
<code>sudo apt-get install tshark</code>	Installs the latest Tshark package, or specifies a previous version number
<code>sudo apt-get upgrade tshark</code>	Upgrades to the latest Tshark package

Wireshark

Wireshark is a GUI-based packet capture utility that is used for packet capture by means of libpcap. The following table lists useful command options.

Table 25: Wireshark commands

Command	Description
<code>sudo apt-get update</code>	Updates any dependency packages
<code>sudo apt-get install wireshark</code>	Installs the latest Wireshark package, or specifies a previous version number
<code>sudo apt-get upgrade wireshark</code>	Upgrades to the latest Wireshark package

Assigning a static IP address on the TPVM Linux OS

To use a static IP address, you add the static method for the interface in the file `/etc/network/interfaces`.

1. Obtain your current address, network mask, and broadcast address.

```
device# ifconfig
...
eth0 Link encap:Ethernet HWaddr 00:0a:21:ff:45:2a
      inet addr:10.10.10.0 Bcast:10.10.10.255 Mask:255.255.255.0
      ...
```

This example displays the first Ethernet interface identified as eth0.

2. Obtain your gateway and network address.

```
device# route -n
Kernel IP routing table
  Destination  Gateway      Genmask      Flags  Metric  Ref  Use  Iface
  0.0.0.0      10.10.10.2   0.0.0.0      UG     100    0    0    eth0
  172.16.77.0  0.0.0.0     255.255.255.0 U       0      0    0    eth0
```

Use flags **u** and **g** for the route gateway. The other IP address is the network IP address.

3. Open the interfaces file.

```
device# sudo nano /etc/network/interfaces
```

Nano is the GNU version of the Pico text editor. Use the editor of your choice.

4. Find the DHCP settings in the `/interfaces` file. They will appear as text similar to the following example.

```
...
auto eth0
iface eth0 inet dhcp
...
```

5. Replace the settings shown in step 4 with the following settings.

```
...
auto eth0
iface eth0 inet static
address 10.0.0.100
netmask 255.255.255.0
```

```
gateway 10.0.0.0
...
```

This example configures the first Ethernet interface, identified as eth0.

6. Save the file and exit to the command prompt.
7. Make sure that your name server IP address is your gateway IP address.

```
device# sudo nano /etc/resolv.conf
```

8. Restart the networking components.

```
device# sudo /etc/init.d/networking restart
```

9. If you want this as a permanent change, remove the DHCP client so it can no longer assign dynamic IP addresses.

```
device# sudo apt-get remove dhcp-client
```

10. Verify connectivity.

```
device# ping www.extremenetworks.com
```

11. Manually enable the interface.

```
device# sudo ifup eth0
```

TPVM on the SLX 9150 series

This section addresses TPVM behavior on the SLX 9150 series platforms.

The SLX 9150 series platforms support only one disk of 128 GB. 64 GB of the disk are used to store SLX-OS, and the remaining 64 GB are used to store the TPVM image and any additional TPVM virtual disks. The TPVM disk image (or TPVM main disk) and any additional TPVM virtual disks share the same single 64-GB partition reserved for TPVM. It is important to be aware of this when creating virtual disks inside the TPVM.

The total size of the TPVM main disk and all the virtual disks is limited to 64 GB, the size of the actual physical partition. Because these platforms are bare-metal systems, the SLX-OS reload behavior affects TPVM. When SLX-OS is rebooted, TPVM is rebooted as well. However, on the contrary, when TPVM is rebooted, SLX-OS is not affected. Because the SLX 9150 series platforms do not support a host OS, the console toggle key sequences “CTRL + y + 1|2|3” is not supported. Instead, a new command, **tpvm console**, allows connection to the TPVM console from an SLX Telnet or console session, as in the following example.

```
[SLX]# tpvm console
Connected to domain TPVM
Escape character is ^\
Ubuntu 16.04.4 LTS TPVM ttyS0
TPVM login:
```

Once in the TPVM console, you can execute **ctrl+\<** to switch back to the session from where the TPVM console was started.



Network Time Protocol (NTP)

[Network Time Protocol overview](#) on page 118

[Configuring NTP](#) on page 122

[Authenticating an NTP server](#) on page 123

[Displaying the active NTP server](#) on page 124

Network Time Protocol overview

Network Time Protocol (NTP) maintains uniform time across all devices in a network. The NTP commands support the configuration of an external time server to maintain synchronization among all local clocks in a network.

To keep the time in your network current, it is recommended that each device have its time synchronized with at least one external NTP server.

Date and time settings

Extreme devices maintain the current date and time inside a battery-backed real-time clock (RTC) circuit. Date and time are used for logging events. Device operation does not depend on the date and time; a device with incorrect date and time settings can function correctly. However, because the date and time are used for logging, error detection, and troubleshooting, you should set them correctly.

Time zone settings

The time zone settings have the following characteristics:

- The setting automatically adjusts for Daylight Savings Time.
- Changing the time zone on a device updates the local time zone setup and is reflected in local time calculations.
- By default, all devices are in the Greenwich Mean Time (GMT) time zone (0,0).
- System services that have already started will reflect the time zone changes only after the next reboot.
- Time zone settings persist across failover for high availability.
- Time zone settings are not affected by NTP server synchronization.

Network Time Protocol Server Overview

The Network Time Protocol (NTP) server provides the correct network time on your device. NTP is used to synchronize the time on devices across a network.

The Network Time Protocol server is used to obtain the correct time from an external time source and adjust the local time in each connected device. When NTP server functionality is enabled, the NTP server starts listening on the NTP port for client requests and responds with the reference time. Up to eight server addresses can be configured in IPv4 or IPv6 format. When multiple NTP server addresses are configured, the NTP algorithm finds the most reliable server and uses this as the active NTP server. If there are no reachable time servers, then the local device time becomes the default time until a new active time server is configured. If an NTP server loses synchronization, it will operate in master mode to serve time using the local clock. Use the **ntp master** command to enable the serving of local time.

The NTP server is stateless and does not maintain NTP client information. Network time synchronization is guaranteed only when a common external time server is used by all devices.



Important

Although time-stepping corrects a large offset after a reload, as a best practice do not manually change the time after NTP synchronization.

Network Time Protocol Client Overview

An NTP client can be enabled when one or more NTP servers/peers is configured.

The NTP client maintains the server and peer state information as an association. The server and peer association is mobilized at startup, or after it has been configured. A statically configured server/peer association is not demobilized unless the configuration is removed/changed. A symmetric passive association is mobilized upon the arrival of an NTP packet from a peer which is not statically configured. This type of association is demobilized on error or timeout.

The NTP client operation can be summarized as follows:

1. The device is booted and the system initializes. The configured servers and peers are polled at the configured poll interval. Additional dynamically discovered servers/peers are also polled.
2. Multiple samples of server/peer times in the NTP packet are added to and maintained in the association database.
3. The selection, cluster, and combine algorithms choose the most accurate and reliable server/peer as system peer.



Note

Refer to RFC 5905.

4. The reference time from the system peer is used for system time synchronization.
5. The NTP client increases the poll interval from the minimum poll interval to the maximum poll interval value after the clock stabilizes.

After the system peer is chosen, the system time is synchronized using one of the following ways:

- If the system time differs from the system peer by less than 128 milliseconds, then the system clock is adjusted slowly towards the system peer time reference time.

- If the system time differs from the system peer by greater than 128 milliseconds, then the system clock is stepped to the system peer reference time. The old, time-related information stored in the server/peer association database is cleared.

Network Time Protocol Associations

NTP works in one or more association modes.

The following modes are the NTP polling based associations:

1. NTP server
2. NTP client
3. NTP peer

NTP Server

The Server mode requires no prior client configuration; it responds to Client mode NTP packets. The **ntp server enable** command is used to set the device to operate in Server mode. Use **no ntp disable serve** to ensure NTP is configured in server mode.

NTP Client

When the system is operating in Client mode, all configured NTP servers and peers are polled. The device selects a host from all the polled NTP Servers from which to synchronize. To configure the NTP servers and peers individually, use the **server** and **peer** commands.

NTP Peer

NTP Peer mode is intended for configurations where a group of devices operate as mutual backup for one another. If one device loses a reference source, the time values flow from the remaining peers.

The NTP peer can operate in:

- **Symmetric Active** - When the peer is configured using the peer command.
- **Symmetric Passive** - If the device is not configured using the peer command, the arrival of an NTP packet from a symmetric active peer generates a symmetric passive response. However, to prevent false time values being introduced, authentication in symmetric mode is strongly suggested.

Network Time Protocol Authentication

NTP can be configured to provide cryptographic authentication of messages with the clients/peers and with the upstream time server.

NTP supports symmetric key scheme for authentication. The scheme uses either MD5 or SHA1 authentication algorithms. The key-id and the calculated digest form the Message Authentication Code (MAC). When authentication is enabled on the server, it is expected that the client's request message has a valid MAC. If authentication of the client message fails, NTP replies with a crypto-NAK packet.

Enabling NTP authentication

To enable NTP strict authentication, use the `authenticate` command. To disable the function, use the `no` form of this command.

```
device(config)# ntp authenticate
```

Syntax: `[no] ntp authenticate`

Defining an authentication key

To define an authentication key for NTP, use the `authentication-key` command. To remove the authentication key for NTP, use the `no` form of this command.

```
device(config)# ntp authentication-key 10 sha1 teststring encryption-level 0
```

Full Syntax: `[no] ntp authentication-key <key-id> <Auth-Type sha1/md5> <Auth-String> encryption-level <0/7>`

The valid key-id parameter is 1 to 65535.

Key type is either SHA1 or MD5. SHA1 specifies message authentication support provided using SHA1 algorithm; MD5 uses the Message Digest 5 Algorithm.

Auth String; secret key string.

Encryption level 0/7; 0 is clear text, 7 is encrypted text.

NTP Trusted Keys

Trusted keys are a set of keys within the set of configured keys used to synchronize a device to a trusted server, and prevent synchronization with a non-trusted device. While it is possible to synchronize a server to a client with only an Authentication key, synchronizing a client to a server requires that an NTP Authentication is enabled on both the client and server, and the same trusted keys be specified on each device. The keys configured for server/peer are implicitly considered trusted keys.



Note

To add a key as trusted key, it must first be configured as an authentication-key.

```
device(config)# [no] ntp trusted-key 10 20
```

Full syntax: `[no] ntp trusted-key <key-id-list>`

Key-id: The allowed range is 1-65535.

A maximum of 10 trusted keys can be configured, and must be configured under the `ntp authentication-key` command.

Configuring NTP

After setting the date and time on a device, the local time on a device can be synchronized with an Network Time Protocol (NTP) server.

The date and time are set in privileged EXEC mode and only have to be configured once per device because the value is written to nonvolatile memory. After the basic time information is set up, an NTP server is configured to allow the local time to be synchronized across the network.

1. Set the current date and time in the UTC timezone for the device.



Note

This MUST be done in the UTC timezone. Otherwise issues will arise as NTP attempts to sync to the upstream servers and peers, and the clock timezone command will incorrectly adjust the time.

```
device# clock set 2016-08-06T12:15:00
```

2. Access global configuration mode.

```
device# configure terminal
```

3. Set the time zone for the device.

```
device(config)# clock timezone America/Los_Angeles
```

4. Return to privileged EXEC mode.

```
device(config)# exit
```

5. Display the local date, time, and time zone for the device.

```
device# show clock
2017-02-09 12:15:00 America/Los_Angeles
```

6. Enter global configuration mode.

```
device# configure terminal
```

7. Synchronize the local time with an external source accessible from a user-specified VRF named myvrf.

```
device(config)# ntp server 192.168.10.1 use-vrf myvrf
```

8. Exit to global configuration mode.

```
device(config)# exit
```

9. Exit to privileged EXEC mode.

```
device(config)# exit
```

10. Display the active NTP server IP address.

```
device# show ntp status
Clock is synchronized, stratum 3, reference clock is 192.168.128.5
precision is 2**24
reference time is CC38EC6A.8FCCA1C4 (10:10:02.561 JST Fri Jan 20 2017 )
clock offset is -1.051 msec, root delay is 174.060 msec
root dispersion is 172.37 msec, peer dispersion is 0.10 msec
system poll interval is 32, last update was 19 sec ago
NTP server mode is enabled, NTP client mode is enabled
NTP master mode is disabled, NTP master stratum is 8
```

In the following example, the date, time and time zone are set on a device and verified. The local device is configured to synchronize the local time with an external NTP server at a specific IP address, accessible from a user-specified VRF named myvrf.

```
device# clock set 2017-02-09 12:15:00
device# configure terminal
device(config)# clock timezone America/Los_Angeles
device(config)# exit
device# show clock
2017-02-09 12:15:00 America/Los_Angeles
device# configure terminal
device(config)# ntp server 192.168.10.1 use-vrf myvrf
device(config)# exit
device(config)# exit
device# show ntp status
Clock is synchronized, stratum 3, reference clock is 192.168.128.5
precision is 2**24
reference time is CC38EC6A.8FCCA1C4 (10:10:02.561 JST Fri Jan 20 2017 )
clock offset is -1.051 msec, root delay is 174.060 msec
root dispersion is 172.37 msec, peer dispersion is 0.10 msec
system poll interval is 32, last update was 19 sec ago
NTP server mode is enabled, NTP client mode is enabled
NTP master mode is disabled, NTP master stratum is 8
```

Authenticating an NTP server

An authentication key can be created for the purpose of authenticating an external Network Time Protocol (NTP) server.

This task demonstrates how to create an authentication key and associate the key to an NTP server.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create an authentication key ID and key string.

```
device(config)# ntp authentication-key 33 md5 check
```

Up to five NTP authentication keys can be configured and each key ID must be unique.

3. Synchronize the local time with an external source, an NTP server, accessible by the management VRF. Associate the key to the NTP server.

```
device(config)# ntp server 192.168.10.1 key 33
```

4. Exit to global configuration mode.

```
device(config)# exit
```

5. Exit to privileged EXEC mode.

```
device(config)# exit
```

In the following example, an authentication key with an ID of 33 is created and the local time on the device is synchronized with an external NTP server at the IP address of 192.168.10.1.

```
device# configure terminal
device(config)# ntp authentication-key 33 md5 check
device(config)# ntp server 192.168.10.1
device(config)# server-192.168.10.1 key 33
device(config)# exit
device(config)# exit
device(config)# ntp authenticate
```

Displaying the active NTP server

Information about the currently active NTP server can be displayed. When an NTP server has been configured, the server IP address is displayed. If an NTP server is not configured or the server is unreachable, the output displays LOCL (for local device time).

Only the local NTP server information is displayed.

NTP server status when an NTP server is not configured

When an NTP server is not configured, the device will work with local time and the NTP status will display as shown below:

```
device# show ntp status
Clock is unsynchronized, no reference clock.
NTP server mode is enabled, NTP client mode is enabled
NTP master mode is disabled
```

NTP server status when an NTP server is configured

The following example shows the status of a configured NTP server:

```
device# show ntp status
Clock is synchronized, stratum 3, reference clock is 192.168.128.5
precision is 2**24
reference time is CC38EC6A.8FCCA1C4 (10:10:02.561 JST Fri Jan 20 2017 )
clock offset is -1.051 msec, root delay is 174.060 msec
root dispersion is 172.37 msec, peer dispersion is 0.10 msec
system poll interval is 32, last update was 19 sec ago

NTP server mode is enabled, NTP client mode is enabled
NTP master mode is disabled, NTP master stratum is 8
```



SNMP

[SNMP overview on page 125](#)

[Configuring SNMPv2 on page 129](#)

[Configuring SNMPv3 on page 130](#)

[Configuring an SNMP server context to a VRF on page 131](#)

[Offline SNMP ifIndex generation tool on page 132](#)

SNMP overview

Simple Network Management Protocol (SNMP) is a set of application layer protocols for managing complex networks. Devices within a network use SNMP to send messages, called protocol data units (PDUs), to different parts of a network.

Network management using SNMP requires three components:

- **SNMP manager**—Typically, network management systems (NMS) that manage networks by monitoring the network parameters, and optionally, setting parameters in managed devices. The SNMP manager communicates to the devices within a network using the SNMP protocol.
- **SNMP agent**—Software that resides in the managed devices in the network, and collects and stores data from these devices. Each device hosts an SNMP agent. The agent receive requests from the SNMP manager and responds with the requested data. In addition, the agent can asynchronously alert the SNMP manager about events by using special PDUs called traps.

Multiple instances of the same MIB module can support a single SNMP agent by mapping a specific key called a context name to a virtual routing and forwarding (VRF) instance created within the Extreme device.

- **Management Information Base (MIB)**—Hierarchical database where SNMP agents in the managed devices store the data about these devices. The MIB is structured on the standard specified in the RFC 2578 [Structure of Management Information Version 2 (SMIv2)].

An SNMP manager can issue read or write operations to retrieve and use the MIB objects to manage and monitor devices on the network. However, the MIB structure determines the scope of management access allowed by a device.

The SNMP server on the Extreme device supports SNMP version 1 (SNMPv1), SNMP version 2 (SNMPv2), and SNMP version 3 (SNMPv3).

- **SNMPv1 and SNMPv2** use community strings associated to SNMP groups. The group maps the user to MIB objects called SNMP views. The views restrict the access of the MIB OIDs .
- **SNMPv3** provides additional security through authenticated users associated with groups to restrict the access of MIBs for SNMP requests through SNMP views.

Also, the device supports the configuration of trap hosts as a trap recipient to receive filtered traps based on their severity level, and optionally receive SNMP communication through a VRF.

When clear command is issued to clear interface statistics, counters are cleared only from CLI version of the statistics and the SNMP version of the statistics are kept intact (SNMP stats preservation). SNMP accumulates the counters and displays aggregate values via IF-MIB queries. These MIB statistics can be preserved by using the **snmp-server preserve-statistics** command by enabling or disabling these MIB statistics when the **clear interface statistics** command is issued.

snmp-server preserve-statistics command is enabled, SNMP MIB statistics are preserved .i.e **clear** command only clears counters from command line interface and not from SNMP IF-MIB. When **snmp-server preserve-statistics** is disabled, **clear** command deletes the counters from both the command line interface and SNMP versions.



Note

By default, preserving of MIB statistics is enabled. User has to execute the CLI command to disable preserving of MIB statistics.

```
device(config)# snmp-server preserve-statistics disable
device(config)# no snmp-server preserve-statistics disable
```

Basic SNMP operation

Every Extreme device carries an *agent* and management information base (MIB), as shown in the next figure. The agent accesses information about a device and makes it available to an SNMP network management station.

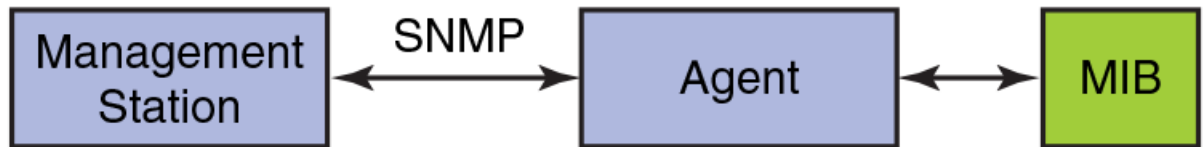


Figure 10: SNMP structure

When active, the management station can "get" information or "set" information when it queries an agent. SNMP commands, such as **get**, **set**, **getnext**, and **getresponse**, are sent from the management station, and the agent replies once the value is obtained or modified as shown in the next figure. Agents use variables to report such data as the number of bytes and packets in and out of the device, or the number of broadcast messages sent and received. These variables are also known as managed objects. All managed objects are contained in a MIB.

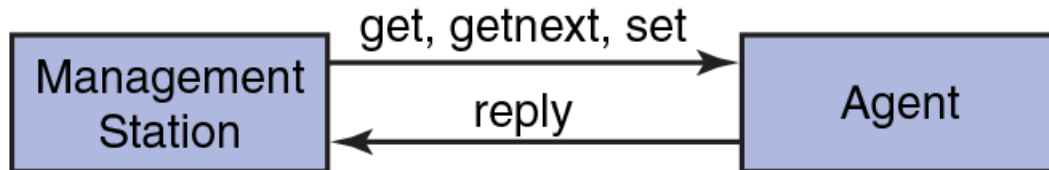


Figure 11: SNMP query

The management station can also receive *traps*, unsolicited messages from the device agent if an unusual event occurs as shown in the next figure.



Figure 12: SNMP trap

The agent can receive queries from one or more management stations and can send traps to up to six management stations.

SNMP community strings

SNMP versions 1 and 2 use community strings to restrict SNMP access.

The community string can be associated with an SNMP group to restrict the access of MIBs for SNMPv1 and SNMPv2c requests. You can configure a total of 256 read-only and read-write community strings on the device.

The software automatically encrypts SNMP community strings. Users with read-only access or who do not have access to management functions in the CLI cannot display the strings. For users with read-write access, the strings are encrypted in the CLI.

By default, you cannot perform any SNMP Set operations until you configure a read-write community string.

SNMP groups

SNMP groups map the SNMP user for SNMPv3 and the community for the SNMPv1 and SNMPv2 to SNMP views.

You can configure each group with any or all of the following views:

- Read view with read-only access
- Write view with read-write access
- Notify view to filter notifications to be encrypted and sent to target hosts

SNMP users that are mapped to a group with SNMP views use its views for access control.

SNMP users

SNMP version 3 (RFC 2570 through 2575) introduces a User-Based Security model (RFC 2574) for authentication and privacy services. This model provides a user that is associated with security information for authentication of its generated SNMP messages.

SNMP version 3 also supports View-Based Access Control Mechanism (RFC 2575) to control access at the PDU level. It defines mechanisms for determining whether to allow access to a managed object in a

local MIB by a remote principal. You can create and associate SNMPv3 users with configured SNMP groups to use the group views for access control.

SNMP views

SNMP views are named groups of MIB objects that you can associate with groups to limit access by community strings and users for viewing and modifying the SNMP statistics and system configuration. With SNMP views, you can create or remove the access to a MIB object for inclusion or exclusion from viewing from user access.

SNMP views reference MIB objects using object names. It represents the hierarchical location of the object in the MIB tree. You associate the views with each group to restrict or allow access to the OIDs. You can create a maximum of 10 views on the device.

SNMP server hosts

On the Extreme device, the SNMP server host serves as a trap receiver to ensure that all SNMP traps sent by the device go to the same SNMP trap receiver or set of receivers, typically one or more host devices on the network.

For an SNMPv3 trap, you associate a SNMPv3 host with the SNMP users. When you specify the host, you also specify a community string for SNMPv1 and SNMPv2. The Extreme device sends all the SNMP traps to the specified hosts and includes the specified community string. Then, administrators can filter for traps from a Extreme device based on IP address or community string.

Multiple SNMP server context to VRF mapping

A single SNMP agent can support multiple instances of the same MIB module by the mapping of the context name to a virtual routing and forwarding (VRF) instance created within the device.

You map each VRF with a specific context name. The context name identifies the VRF and fetches the MIB details of the mapped VRF from the underlying modules. For example, the OSPF-MIB returns the queried OSPF-MIB object values pertaining to the default VRF (default-vrf).

For SNMPv1 and SNMPv2, the mapping of the context is with the community. This mapping is in addition to mapping of the context with the VRF. The SNMP agent supports 256 contexts to support context-to-VRF mapping.

For SNMPv3, you only need to map the context with the VRF. The SNMPv3 request PDU itself provisions for the context. Only one context is allowed for each VRF instance.

SNMP source interface

The SNMP source interface uses the IP address of the configured interface loopback, virtual interface or management IP as the source IP address for SNMP trap and inform packets originated from the device.

The specified interface acts as the source interface for SNMP trap and inform the packets. SNMP trap host can be configured for SNMP version 1, version 2, and version 3 per instance. If the source interface is not specified, the source IP address is the IP address of the interface through which packet exits device. If the source interface is modified (changing IP address), then it is reflected in the trap packets.

Configured source interface IP address is not cached because the corresponding IP address can be modified. While sending out the SNMP trap packets to find the source IP address to use, the system checks and picks up the source interface configured. If an interface with no IP address is configured as the source interface, SNMP trap packets have the egress interface IP as the source IP.

The SNMP source interface supports the following interface types:

- Virtual routing interface
- Loopback interface
- Management IP

The Source interface configurations are stored in the running-config and can be viewed using command name **show running-config**.

```
device# show running-config

snmp-server host 192.168.12.5 $9$U13Qs06hXODXilnToF/r9Q==
source-interface loopback 1
```

Configuring SNMPv2

SNMPv1 and SNMPv2 use community strings to restrict SNMP access. When you associate it with an SNMP group, you can restrict the access of MIBs for SNMPv1 and SNMPv2c requests.

To configure SNMPv2, perform the following steps.

1. In privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Add an SNMP view to restrict or allow access to the MIB OIDs.

```
device(config)# snmp-server view view2 1.3.6.1 included
```

This example adds the SNMP view2 view entry with included permission to allow access for the MIB 1.3.6.1 object ID ('internet').

3. Add an SNMP group.

```
device(config)# snmp-server group admin v2c write view2 notify view2
```

This example adds the admin group for SNMPv2 and maps the read-write access and notify views to view2.

4. Add an SNMP community string and associate it with a group.

```
device(config)# snmp-server community comm1 group admin
```

This example adds the comm1 community string and associates it with the admin group to access the MIBs for SNMPv2c requests.

5. Configure the SNMP trap host associated with community string.

```
device(config)# snmp-server host 10.32.147.6 comm1 version 2c severity-level Warning
```

This example configures 10.32.147.6 as a trap recipient with SNMPv2c on the default target port 162 and associates the comm1 community string.

6. Enable the traps.

```
device(config)# snmp-server enable trap
```

7. Access privileged EXEC mode.

```
device(config)# exit
```

8. Verify the configuration.

```
device# show running-config snmp-server
snmp-server contact "Operator 12345"
snmp-server enable trap
snmp-server location "Building 3 Room 214"
snmp-server community comm1 group admin
snmp-server group admin v2c write view2 notify view2
snmp-server host 10.32.147.6 comm1 version 2c
severity-level Warning
!
```

The following example shows the previous steps to configure SNMPv2.

```
device# configure terminal
device(config)# snmp-server location "Building 3 Room 214" contact "Operator 12345"
device(config)# snmp-server view view2 1.3.6.1 included
device(config)# snmp-server group admin v2c write view2 notify view2
device(config)# snmp-server community comm1 group admin
device(config)# snmp-server host 10.32.147.6 comm1 version 2c severity-level Warning
device(config)# snmp-server enable trap
```

Configuring SNMPv3

SNMPv3 uses SNMP users to restrict SNMP access. When you map an SNMP user to an SNMP group, you can restrict the access of MIBs for SNMP requests through an SNMP view.

To configure SNMPv3, perform the following steps.

1. In privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Configure the contact information for the SNMP server.

```
device(config)# snmp-server contact "Network Management group - Contact # 123-123-1234"
```

This example changes the default contact information from Field Support to "Network Management group - Contact # 123-123-1234".

The double quotes allows you to enter the string with spaces.

3. Configure the location information for the SNMP server.

```
device(config)# snmp-server location "South Room, Rack-11"
```

This example changes the default location from End User Premise to "South Room, Rack-11".

The double quotes allows you to enter the string with spaces.

4. Add an SNMP view to restrict or allow access to the MIB OIDs.

```
device(config)# snmp-server view view2 1.3.6.1 included
```

This example adds the SNMP view2 view entry with included permission to allow access for the MIB 1.3.6.1 object ID ('internet').

5. Add an SNMP group.

```
device(config)# snmp-server group group1 v3 priv write view2 notify view2
```

This example adds the group1 group for SNMPv3 and maps the read-write access and notify views to view2.

6. Add an SNMP user and associate it with a group.

```
device(config)# snmp-server user user2 groupname group1 auth md5 auth-password
private123 priv DES priv-password public123
```

This example adds the user2 user and associates it with the group1 group to access of MIBs for SNMPv3 requests. For SNMPv3 users, the passwords for **auth-password** and **priv-password** keywords are encrypted while storing to the persistent memory or displaying it back to the user. You can configure either with a plain-text password or an encrypted password. In both cases, the **show running-config** command displays the passwords as encrypted.

7. Configure the SNMPv3 trap host associated with an SNMP user.

```
device(config)# snmp-server v3host 10.26.3.166 user2 severity-level Info udp-port
4425
```

This example configures 10.26.3.166 as an SNMPv3 trap recipient host on the target port 4425 and associates the user2 user.

The global SNMPv3 host can be associated with global SNMPv3 users only. You cannot create an SNMPv3 host in a global configuration by associating it with local SNMPv3 users.

8. Enable the traps.

```
device(config)# snmp-server enable trap
```

9. Access privileged EXEC mode.

```
device(config)# exit
```

10. Verify the configuration.

```
device# show running-config snmp-server
snmp-server contact "Network Management group - Contact # 123-123-1234"
snmp-server enable trap
snmp-server location "South Room, Rack-11"
snmp-server group group1 v3 priv write view2 notify view2
snmp-server user user2 groupname group1 md5 auth-password private123 priv
password public123
snmp-server v3host 10.26.3.166 user2
severity-level Info
udp-port 4425
!
snmp-server view view2 1.3.6.1 included
```

The following example shows the previous steps to configure SNMPv3.

```
device# configure terminal
device(config)# snmp-server contact "Network Management group - Contact # 123-123-1234"
device(config)# snmp-server location "South Room, Rack-11"
device(config)# snmp-server view view2 1.3.6.1 included
device(config)# snmp-server group group1 v3 priv write view2 notify view2
device(config)# snmp-server user user2 groupname group1 md5 auth-password private123 priv
DES priv-password public123
device(config)# snmp-server v3host 10.26.3.166 user2 severity-level Info udp-port 4425
device(config)# snmp-server enable trap
```

Configuring an SNMP server context to a VRF

A single SNMP agent can support multiple instances of the same MIB module by mapping the context name to a virtual routing and forwarding (VRF) instance created within the device. The SNMP context

name is used to identify the VRF and fetch the MIB details of the mapped VRF from the underlying modules.

To configure an SNMP server context to a VRF for SNMPv1 or SNMPv2, perform the following steps.



Note

For SNMPv3, use the **snmp-server context** command only. The SNMPv3 request PDU itself has the provision for the context name as input.



Important

SNMP SET requests work only on the default VRF.

1. In privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Map the context with the community.

```
device(config)# snmp-server community public groupname admin
```

3. Create a context and map it with a VRF.

```
device(config)# snmp-server context mycontext vrf myvrf
```

4. Map the community to the context.

```
device(config)# snmp-server mib community-map public context mycontext
```

5. Verify the configuration.

```
device# show running-config snmp-server
...
snmp-server community public groupname admin
snmp-server context mycontext vrf myvrf
...
snmp-server mib community-map public context mycontext
```

The following example shows the previous steps for the configuration.

```
device# configure terminal
device(config)# snmp-server community public groupname admin
device(config)# snmp-server context mycontext vrf myvrf
device(config)# snmp-server mib community-map public context mycontext
```

Offline SNMP ifIndex generation tool

On Extreme SLX Router, SNMP Management Information Base (MIB) uses Interface Index (ifIndex) to assign a unique identifying value to each interface.

The ifIndex is encoded per interface type and the assigned value is used if any information needs to be polled for a particular interface. The offline SNMP ifIndex generation tool which is developed based on Python, provides a means to find out the ifIndexes associated with various interfaces. This tool can run on any platform (SLX Router, Linux, or Windows) wherever the Python package is installed. The script is available on SLX Router at `/fabos/cliexec/ifindex_gen.py`. If you want to run it on a Linux or Windows platform, you may have to modify the first line in the script to point to the location of the Python binary on the platform.

Generating ifIndexes for various interfaces

ifIndex can be generated offline for various interface types such as physical interface, LAG (port-channel) interface, VE interface, loopback interface, tunnel interface and Management interface.

To generate ifIndex for a specific interface, perform the following steps.

1. Enter SLX-OS Linux shell from privileged EXEC mode.

```
device# start-shell
```

2. Retrieve syntax to find available options to generate ifIndex.

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -h
usage: ifindex_gen.py [-h] -i INTF_TYPE [-t LC_TYPE] [-m MODE] [-s SLOT]
                    [-p PORT] [-sp SUB_PORT] [-lp LAG_PORT]
                    [-vb VE_BRIDGE_ID] [-vi VE_INTF_ID] [-tt TUNNEL_TYPE]
                    [-ti TUNNEL_ID] [-lbi LB_INTF_ID] [-mi MGMT_INTF_ID]
                    [-d DISP_MODE]

arguments:
-h, --help            show this help message and exit
-i INTF_TYPE          Interface type: [phy (physical) | lag | ve | tunnel | lb
                    (loopback) | mgmt (management)]
-t LC_TYPE            LC type: [72x10G | 36x100G]
-m MODE              PortGroup Mode: [40g | 100g] (required when LC type is
                    36x100G)
-s SLOT              Slot #: [1-8]
-p PORT              Port #: [1-72] for 72x10G, [1-60] for 36x100G LC type
-sp SUB_PORT          Sub Port #: [1-4] for break-out ports (required when LC
                    type is 36x100G, PortGroup Mode is 40g and break-out is
                    enabled)
-lp LAG_PORT          LAG Port #: [1-512]
-vb VE_BRIDGE_ID     VE Bridge ID: [0-255]
-vi VE_INTF_ID       VE Interface ID: [1-4096]
-tt TUNNEL_TYPE      Tunnel type: [vxlan | gre | nvgre | mpls]
-ti TUNNEL_ID        Tunnel ID: [1-1024]
-lbi LB_INTF_ID      Loopback Interface ID: [1-255]
-mi MGMT_INTF_ID     Management Interface ID: [1-2]
-d DISP_MODE         Output Display Mode: [bin | dec | hex | all] (default:
                    dec)
```

Note: The parameters -t, -m, -s, -p, and -sp are the sub-options specific to physical interface.

3. Generate ifIndex for a specific interface. In this example, ifIndex is generated for a physical interface.

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i physical -t 72x10G -s 2 -p 65 -d all
Decimal : 413171855
Hex      : 18a0808f
Binary   : 00011000101000001000000010001111
```

Configuration examples for generating ifIndexes offline

The following examples provide details on how ifIndexes can be generated for various interface types.

Physical interfaces with LC type 72x10G

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i physical -t 72x10G -s 2 -p 65 -d all
Decimal : 413171855
Hex      : 18a0808f
Binary   : 00011000101000001000000010001111
```

Physical interfaces with LC type 36x100G (100g mode)

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i phy -t 36x100G -m 100g -s 3 -p 1 -d all
Decimal : 415285249
```

```
Hex      : 18c0c001
Binary   : 0001100011000000011000000000000001
```

Physical interfaces with LC type 36x100G (40g mode) non-breakout

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i phy -t 36x100G -m 40g -s 3 -p 8 -d all
Decimal  : 207683777
Hex      : 0c6100c1
Binary   : 00001100011000010000000011000001
```

Physical interfaces with LC type 36x100G (40g mode) breakout

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i phy -t 36x100G -m 40g -s 3 -p 15 -sp 1 -d
all
Decimal  : 207741442
Hex      : 0c61e202
Binary   : 00001100011000011110001000000010
```

LAG (Port-channel) interface

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i lag -lp 1 -d all
Decimal  : 671088641
Hex      : 28000001
Binary   : 0010100000000000000000000000000001
```

VE interface

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i ve -vi 10 -d all
Decimal  : 1207959562
Hex      : 4800000a
Binary   : 01001000000000000000000000000001010
```

Tunnel interfaces

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i tunnel -tt mpls -ti 2 -d all
Decimal  : 2092957698
Hex      : 7cc00002
Binary   : 0111110011000000000000000000000010
```

Loopback interface

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i lb -lbi 20 -d all
Decimal  : 1476395028
Hex      : 58000014
Binary   : 010110000000000000000000000000010100
```

Management interface

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i mgmt -mi 2 -d all
Decimal  : 805306370
Hex      : 30000002
Binary   : 0011000000000000000000000000000010
```



LLDP

[LLDP overview](#) on page 135

[Configuring and managing LLDP](#) on page 137

LLDP overview

The IEEE 802.1AB Link Layer Discovery Protocol (LLDP) enhances the ability of network management tools to discover and maintain accurate network topologies and simplify LAN troubleshooting in multi-vendor environments. To efficiently and effectively operate the various devices in a LAN you must ensure the correct and valid configuration of the protocols and applications that are enabled on these devices. With Layer 2 networks expanding dramatically, it is difficult for a network administrator to statically monitor and configure each device in the network.

Using LLDP, network devices such as routers and switches advertise information about themselves to other network devices and store the information they discover. Details such as device configuration, device capabilities, and device identification are advertised. LLDP defines the following:

- A common set of advertisement messages.
- A protocol for transmitting the advertisements.
- A method for storing the information contained in received advertisements.



Note

LLDP runs over the data-link layer which allows two devices running different network layer protocols to learn about each other.

LLDP information is transmitted periodically and stored for a finite period. Every time a device receives an LLDP advertisement frame, it stores the information and initializes a timer. If the timer reaches the time to live (TTL) value, the LLDP device deletes the stored information ensuring that only valid and current LLDP information is stored in network devices and is available to network management systems.

Layer 2 topology mapping

The LLDP protocol lets network management systems accurately discover and model Layer 2 network topologies.

As LLDP devices transmit and receive advertisements, the devices store information they discover about their neighbors. Advertisement data such as a neighbor's management address, device type, and port identification is useful in determining what neighboring devices are in the network.



Note

The Extreme LLDP implementation supports up to two neighbors.

The higher level management tools, such as the Network Advisor, can query the LLDP information to draw Layer 2 physical topologies. The management tools can continue to query a neighboring device through the device's management address provided in the LLDP information exchange. As this process is repeated, the complete Layer 2 topology is mapped.

In LLDP the link discovery is achieved through the exchange of link-level information between two link partners. The link-level information is refreshed periodically to reflect any dynamic changes in link-level parameters. The basic format for exchanging information in LLDP is in the form of a type, length, value (TLV) field.

LLDP keeps a database for both local and remote configurations. The LLDP standard currently supports three categories of TLVs. The Extreme LLDP implementation adds a proprietary Extreme extension TLV set. The four TLV sets are described as follows:

- Basic management TLV set — This set provides information to map the Layer 2 topology and includes the following TLVs:
 - Chassis ID TLV — Provides the ID for the switch or router where the port resides. This is a mandatory TLV.
 - Port ID TLV—Provides a unique identifiable information of the port. The Port ID could be one of the following: MAC address, Network address, Interface name of the port. On the SLX-OS, the interface name of the port is provided. This is a mandatory TLV.
 - Port description TLV — Provides a description of the port in an alphanumeric format. If the LAN device supports RFC-2863, the port description TLV value equals the "ifDescr" object. This is an optional TLV.
 - System name TLV — Provides the system-assigned name in an alphanumeric format. If the LAN device supports RFC-3418, the system name TLV value equals the "sysName" object. This is an optional TLV.
 - System description TLV — Provides a description of the network entity in an alphanumeric format. This includes system name, hardware version, operating system, and supported networking software. If the LAN device supports RFC-3418, the value equals the "sysDescr" object. This is an optional TLV.
 - System capabilities TLV — Indicates the primary functions of the device and whether these functions are enabled in the device. The capabilities are indicated by two octets. The first octet indicates Other, Repeater, Bridge, WLAN AP, Router, Telephone, DOCSIS cable device, and Station, respectively. The second octet is reserved. This is an optional TLV.
 - Management address TLV — Indicates the addresses of the local switch. Remote switches can use this address to obtain information related to the local switch. This is an optional TLV.

- IEEE 802.1 organizational TLV set — This set provides information to detect mismatched settings between local and remote devices. A trap or event can be reported once a mismatch is detected. This is an optional TLV. This set includes the following TLVs:
 - Port VLANID TLV — Indicates the port VLAN ID (PVID) that is associated with an untagged or priority tagged data frame received on the VLAN port.
 - PPVLAN ID TLV — Indicates the port- and protocol-based VLAN ID (PPVID) that is associated with an untagged or priority tagged data frame received on the VLAN port. The TLV supports a "flags" field that indicates whether the port is capable of supporting port- and protocol-based VLANs (PPVLANs) and whether one or more PPVLANs are enabled. The number of PPVLAN ID TLVs in a Link Layer Discovery Protocol Data Unit (LLDPDU) corresponds to the number of the PPVLANs enabled on the port.
 - VLAN name TLV — Indicates the assigned name of any VLAN on the device. If the LAN device supports RFC-2674, the value equals the "dot1QVLANStaticName" object. The number of VLAN name TLVs in an LLDPDU corresponds to the number of VLANs enabled on the port.
 - Protocol identity TLV — Indicates the set of protocols that are accessible at the device's port. The protocol identity field in the TLV contains a number of octets after the Layer 2 address that can enable the receiving device to recognize the protocol. For example, a device that wishes to advertise the spanning tree protocol includes at least eight octets: 802.3 length (two octets), LLC addresses (two octets), 802.3 control (one octet), protocol ID (two octets), and the protocol version (one octet).
- IEEE 802.3 organizational TLV set — This is an optional TLV set. This set includes the following TLVs:
 - MAC/PHY configuration/status TLV — Indicates duplex and bit rate capabilities and the current duplex and bit rate settings of the local interface. It also indicates whether the current settings were configured through auto-negotiation or through manual configuration.
 - Power through media dependent interface (MDI) TLV — Indicates the power capabilities of the LAN device.
 - Link aggregation TLV — Indicates whether the link (associated with the port on which the LLDPDU is transmitted) can be aggregated. It also indicates whether the link is currently aggregated and provides the aggregated port identifier if the link is aggregated.
 - Maximum Ethernet frame size TLV — Indicates the maximum frame size capability of the device's MAC and PHY implementation.

LLDP configuration guidelines and restrictions

Follow these LLDP configuration guidelines and restrictions when configuring LLDP:

- The Extreme implementation of LLDP supports standard LLDP information.
- Mandatory TLVs are always advertised.
- The exchange of LLDP link-level parameters is transparent to the other Layer 2 protocols. The LLDP link-level parameters are reported by LLDP to other interested protocols.

Configuring and managing LLDP

The following sections discuss working with the Link Layer Discovery Protocol (LLDP) on Extreme devices.

Understanding the default LLDP

The following table lists the default LLDP configuration. Consider this when making changes to the defaults.

Table 26: Default LLDP configuration

Parameter	Default setting
LLDP global state	Enabled
LLDP receive	Enabled
LLDP transmit	Enabled
Transmission frequency of LLDP updates	30 seconds
Hold time for receiving devices before discarding	120 seconds

Disabling LLDP globally

LLDP is enabled globally by default. You can disable LLDP globally without changing any other aspect of the LLDP configuration.

To globally disable LLDP, perform the following steps:

1. From privileged EXEC mode, access global configuration mode.

```
device# configure terminal
```

2. Access LLDP configuration mode.

```
device(config)# protocol lldp
```

3. Disable LLDP globally.

```
device(config-lldp)# disable
```

4. Verify the configuration.

```
device(config-lldp)# do show running-config protocol lldp
protocol lldp
  system-description Extreme BR-SLX9540 Router
  disable
!
```

The following configuration is an example of the previous steps to disable LLDP.

```
device# configure terminal
device(config)# protocol lldp
device(config-lldp)# disable
```

If required, re-enable LLDP.

```
device(config-lldp)# no disable
```

Configuring LLDP global parameters

When LLDP is enabled, the default values of its parameters are set. You can change the configuration of these parameters in LLDP configuration mode.

Specifying a system name for the Extreme device hardware

The global system name for LLDP is useful for differentiating between devices. By default, the host name from the chassis/entity management information base is used. By specifying a descriptive system name, you may find it easier to distinguish the device with LLDP. The following example changes the system name to `Extreme_Alpha`.

```
device(conf-lldp)# system-name Extreme_Alpha
```

Specifying an LLDP system description

The default system description depends on the device type. For example, the default description for an SLX 9540 router is `Extreme SLX9540 Router`.

Extreme recommends that you use the operating system version for the description or use the description from the chassis/entity management information base (MIB).

Do not use special characters, such as `#$!@`, as part of the system name and description. The following example specifies the `IT_1.6.2_LLDP_01` system description.

```
device(conf-lldp)# system-description IT_1.6.2_LLDP_01
```

Specifying a user description for LLDP

A user description for LLDP is for network administrative purposes and is not seen by neighboring devices. The following example specifies the `LLDP-installed-jan-25` description.

```
device(conf-lldp)# description Extreme-LLDP-installed-jan-25
```

Enabling and disabling the receiving and transmitting of LLDP frames

By default both transmit and receive for LLDP frames is enabled.

- The following example enables only receiving of LLDP frames.

```
device(conf-lldp)# mode rx
```

- The following example enables only transmitting of LLDP frames.

```
device(conf-lldp)# mode tx
```

Configuring the transmit frequency of LLDP frames

The default transmit frequency of LLDP frames is 30 seconds. You can change the frequency from 4 to 180 seconds. The following example changes the frequency to 45 seconds.

```
device(conf-lldp)# hello 45
```

Configuring the hold time for receiving devices

By default, four consecutive LLDP hello packets can be missed before removing the neighbor information. You can configure from 1 to 10 consecutive LLDP hello packets that can be missed before removing the neighbor information. The following example configures the 6 consecutive LLDP hello packets.

```
device(conf-lldp)# multiplier 6
```

Advertising the optional LLDP TLVs

By default, the port description and system name are advertised

You can advertise the rest of the optional LLDP TLVs. The following example advertises the management address, capabilities, name and description of the device, and user-configured port.

```
device(conf-lldp)# advertise optional-tlv management-address port-description system-
capabilities system-name system-description
```

Configuring the advertisement of LLDP organizationally-specific TLVs

You have the option of advertising dot1.tlv and dot3.tlv. The following example advertise dot1.tlv.



Note

Extreme does not recommend advertising dot1.tlv and dot3.tlv LLDPs if your network contains Converged Network Adapters (CNAs) from non-Extreme vendors. Functionality problems can occur.

```
device(conf-lldp)# advertise dot1-tlv
```

Configuring LLDP profiles

SLX 9240 supports 128 active profiles and SLX 9140 supports 72 active profiles. When you configure a profile, its default parameters are from the global LLDP configuration.

1. In privileged EXEC mode, access global configuration mode.

```
device# configure terminal
```

2. Enter LLDP configuration mode.

```
device(config)# protocol lldp
```

3. Configure the profile name.

```
device(conf-lldp)# profile UK_LLDP_IT
```

4. Specify a description for the profile.

```
device(conf-lldp-profile-UK_LLDP_IT)#description standard_profile_by_Jane
```

5. Configure the transmission frequency of LLDP updates.

```
device(conf-lldp-profile-UK_LLDP_IT)# hello 10
```

6. Configure the hold time for receiving devices.

```
device(conf-lldp-profile-UK_LLDP_IT)# multiplier 2
```

7. Advertise the optional LLDP TLVs.

```
device(conf-lldp)# advertise optional-tlv system-name
```

8. Advertise the LLDP organizationally-specific TLVs.

```
device(conf-lldp-profile-UK_LLDP_IT)# advertise advertise dot1-tlv
```



Note

Extreme does not recommend advertising dot1.tlv and dot3.tlv LLDPs if your network contains CNAs from non-Extreme vendors. Functionality problems can occur.

9. Return to privileged EXEC mode.

```
device(conf-lldp-profile-UK_LLDP_IT)# end
```

10. Verify the configuration.

```
device# show running-config protocol lldp profile
profile UK_LLDP_IT
hello 10
multiplier 2
advertise dot1-tlv
advertise option-tlv system-name
description standard_profile_by_Jane
```

The following configuration is an example of the previous steps.

```
device# configure terminal
device(config)# protocol lldp
device(conf-lldp)# profile UK_LLDP_IT
device(conf-lldp-profile-UK_LLDP_IT)# description standard_profile_by_Jane
device(conf-lldp-profile-UK_LLDP_IT)# hello 10
device(conf-lldp-profile-UK_LLDP_IT)# multiplier 2
device(conf-lldp-profile-UK_LLDP_IT)# advertise option-tlv system-name
device(conf-lldp-profile-UK_LLDP_IT)# advertise advertise dot1-tlv
```

Configuring an LLDP profile to an interface

You can assign only one LLDP profile to an interface. If you do not use the **lldp profile** option at the interface level, the interface uses the global LLDP configuration.

To configure LLDP interface-level command options, perform the following steps.

1. In privileged EXEC mode, access global configuration mode.

```
device# configure terminal
```

2. Access the interface configuration mode.

```
device(config)# interface Ethernet 0/8
```

3. Apply an LLDP profile to the interface.

```
device(conf-if-eth-0/8)# lldp profile network_standard
```

4. Return to privileged EXEC mode.

```
device(conf-if-eth-0/8)# end
```

5. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

The following configuration is an example of the previous steps.

```
device# configure terminal
device(config)# interface Ethernet 0/8
device(conf-if-eth-0/8)# lldp profile network_standard
```

Displaying LLDP information

The **show lldp** command allows you to display the following information:

- LLDP status
- LLDP neighbor information
- LLDP statistics

Displaying LLDP status

To display the global LLDP status, use the **show lldp** command.

```
device# show lldp
LLDP Global Information
  system-name: SLX
  system-description: Extreme BR-SLX9540-4 Router
  description: Extreme-LLDP
  State:                Enabled
  Mode:                 Receive/Transmit
  Advertise Transmitted: 30 seconds
  Hold time for advertise: 120 seconds
  Tx Delay Timer:      1 seconds
  Transmit TLVs:       Chassis ID          Port ID
                      TTL                Port Description
                      System Name
```

To display LLDP status for an Ethernet interface, use the **show lldp interface ethernet** command.

```
device# show lldp interface ethernet 0/18
LLDP information for Eth 0/18
  State:                Enabled
  Mode:                 Receive/Transmit
  Advertise Transmitted: 30 seconds
  Hold time for advertise: 120 seconds
  Tx Delay Timer:      1 seconds
  Transmit TLVs:       Chassis ID          Port ID
                      TTL                Port Description
                      System Name
```

Displaying LLDP neighbor information

To display the LLDP neighbor information, use the **show lldp neighbors** command. This command allows you to display the information for all Ethernet interfaces, a specific interface, or detailed neighbor information.

The following example displays the LLDP neighbor information for all interfaces.

```
device# show lldp neighbors
Local Port  Dead Interval  Remaining Life  Remote Port ID  Remote Port Descr Chassis
ID          Tx   Rx   System Name
Eth 0/18    120                102            Ethernet 0/25   Eth 2/25
768e.f807.6000 653 652 R6
Eth 0/21    120                108            Ethernet 0/21   Eth 1/21
768e.f807.6000 653 652 R6
Eth 0/40    120                110            Ethernet 0/50   Eth 1/50
768e.f807.6000 653 650 R6
Eth 0/43    120                102            Ethernet 0/51   Eth 2/51
768e.f807.6000 653 652 R6
Eth 0/50    120                102            Ethernet 0/23   Eth 2/23
768e.f807.6000 653 611 R6
```

The following example displays the LLDP neighbor information for Ethernet interface 0/18.

```
device# show lldp neighbors interface ethernet 0/18
Local Port  Dead Interval  Remaining Life  Remote Port ID  Remote Port Descr Chassis
ID          Tx   Rx   System Name
Eth 0/18    120                115            Ethernet 0/25   Eth 0/25
768e.f807.6000 655 654 R6
```

The following example displays the detailed LLDP neighbor information for Ethernet interface 1/18.

```
device# show lldp neighbors interface ethernet 0/18 detail
Neighbors for Interface Eth 0/18

MANDATORY TLVs
=====
Local Interface: Eth 0/18 (Local Interface MAC: 768e.f805.5816)
Remote Interface: Ethernet 0/25 (Remote Interface MAC: 768e.f807.610d)
Dead Interval: 120 secs
Remaining Life : 118 secs
Chassis ID: 768e.f807.6000
LLDP PDU Transmitted: 656 Received: 655

OPTIONAL TLVs
=====
Port Interface Description: Eth 0/25
System Name: R6
```

Displaying LLDP statistics

To display the LLDP statistics for all interfaces or a specific interface, use the **show lldp statistics** command.

The following example displays the statistics for Ethernet interface 0/18.

```
device# show lldp statistics interface ethernet 0/18
LLDP Interface statistics for Eth 0/18
Frames transmitted: 659
Frames Aged out: 0
Frames Discarded: 0
Frames with Error: 0
Frames Recieved: 657
TLVs discarded: 0
TLVs unrecognized: 0
```

If you do not include the **interface ethernet** option, the command displays the statistics for all interfaces.

Clearing LLDP-related information

You can clear LLDP neighbor and statistic information for all interfaces or a specified interface.

To clear LLDP-related information, perform the following steps.

1. In privileged EXEC mode, clear the LLDP neighbor information.

```
device# clear lldp neighbors
```

This example clears the LLDP neighbor information for all interfaces.

2. Clear the LLDP statistics on an interface.

```
device# clear lldp statistics interface ethernet 1/8
```

This example clears the LLDP transmit and receive counters on the Ethernet interface 1/8.

3. Clear the LLDP statistics for all interfaces.

```
device# clear lldp statistics
```



Account and Password Recovery

[Recover the admin password from the root account on page 144](#)
[Root account and password recovery on page 144](#)

Recover the admin password from the root account

If you lose access to the SLX-OS admin account but you have access to the root account for the device, you can recover the password.

Perform the following steps to reset the admin password from the root account.

1. Open a session to access the device.
2. Log in as root.
3. Start the SLX-OS CLI.

```
[root@device]# slxcli  
device#
```

4. Access global configuration mode.

```
device# configure terminal
```

5. Reset the admin password.

```
device(config)# username admin password password
```

In this example, the admin password is reset to the default value of `password`.

You can now use the admin account to manage the admin and user passwords by using normal password-management procedures.

Root account and password recovery

By default, the root account on the virtual machine (VM) is disabled. To log into root, you can log into the SLX-OS CLI and enable the root account from global configuration mode by using **root enable** command. In rare cases, SLX-OS CLI may not be available to enable the root account.

The ability to enable the root account and recover the root credentials (password) depends on the `uboot` environment variable. When the variable is set, it executes the root recovery logic based on the parameter set. The variable is not preserved across reboot. Every time a reboot occurs, the root account is disabled by default and this variable has to be set again to enable it unless the root account was not enabled from global configuration mode.

The root account access availability determines the method for password recovery:

- When the root account is disabled and the SLX-OS CLI is not available, you must recover the root login account. The password is also recovered. Based on your device, perform the relevant task:
 - [\(DNX devices\) Recover the root login account](#) on page 145
 - [\(XGS devices\) Recover the root login account](#) on page 146
- When the root account is enabled but the root password is not available, perform the relevant task:
 - [\(DNX devices\) Recover the root password](#) on page 146
 - [\(XGS devices\) Recover the root password](#) on page 147



Note

The default password for the root account on the VM is `fibranne`.

(DNX devices) Recover the root login account

If the root account is disabled and SLX-OS CLI is not available, to recover the VM root password first recover the VM root login account.



Note

For a list of the currently supported DNX devices, see the "Supported Hardware" topic.

1. Reboot the device.

```
# reboot
Press Esc during reboot.
Hit ESC to stop autoboot: 0
FPGA f6000720 -> 0x12

1) Start system.
2) Recover password.
3) Enter command shell.

Option?
```

2. Choose option 3 to access the uboot prompt.

```
Option? 3
=>
```

The uboot prompt appears.

3. Define the root login value for the root recover environment variable.

```
=> setenv VM_Root_Recover RootLogin
=>
```

This step sets the `VM_Root_Recover` variable with the `RootLogin` value.

4. Save the variable to flash memory.

```
k
=> saveenv
Saving Environment to SPI Flash...
SF: Detected W25Q128BV @ 0:0 with page size 256 Bytes, erase size 64 KiB, 32 KiB, 4
KiB, total 16 MiB
Erasing SPI flash...Writing to SPI flash...
Erasing SPI flash...Writing to SPI flash...done
=>
```

5. Reboot.

```
=> boot
6912784 bytes read in 152 ms (43.4 MiB/s)
```

```
Valid Boot Flag
Setup Size = 0x00004400
Magic signature found
Using boot protocol version 2.0c
Linux kernel version 3.14.17 (raop@hq1-ub-ecbld-373) #1 SMP Thu Jul 7 19:43:15 UTC
2016
```

The root account is now enabled. You can login with the default password.

If the SLX-OS CLI is available, you can recover the root account by using the SLX-OS CLI **root enable** command.

(XGS devices) Recover the root login account

If the root account is disabled and SLX-OS CLI is not available, to recover the VM root password first recover the VM root login account.



Note

For a list of the currently supported XGS devices, see the "Supported Hardware" topic.

1. Enter the **reboot** command.
2. Select and enter the **ONIE** option.
3. Select and enter the **ONIE: Rescue** option.
4. Define the root login value for the root recover environment variable.

```
ONIE:/ # bootenv VM_Root_Recover RootLogin
```

5. Enter the **reboot** command.

```
ONIE:/ # reboot
ONIE:/ # discover: Rescue mode detected. No discover stopped.
Stopping: dropbear ssh daemon... done.
```

The root account is now enabled. You can log in with the default password.

6. If the SLX-OS CLI is available, you can recover the root account by using the SLX-OS CLI **root enable** command.

(DNX devices) Recover the root password

If you forgot the password for the VM root account, you can recover the default password.



Note

To perform the recovery process, you need access to the console prompt.

For VM root password recovery, perform the following steps.

1. Reboot the device.

```
# reboot
Press Esc during reboot.
Hit ESC to stop autoboot: 0
FPGA f6000720 -> 0x12

1) Start system.
2) Recover password.
3) Enter command shell.

Option?
```

- Choose option 3 to access the uboot prompt.

```
Option? 3
=>
```

- Define the root password value for the root recovery environment variable.

```
=> bootenv VM_Root_Recover RootPasswd
=>
```

This step sets the VM_Root_Recover variable with the RootPasswd value.

- Save the variable to flash memory.

```
=> saveenv
Saving Environment to SPI Flash...
SF: Detected W25Q128BV @ 0:0 with page size 256 Bytes, erase size 64 KiB, 32 KiB, 4
KiB, total 16 MiB
Erasing SPI flash...Writing to SPI flash...
Erasing SPI flash...Writing to SPI flash...done
=>
```

- Reboot the device.

```
=> boot
6912784 bytes read in 152 ms (43.4 MiB/s)
Valid Boot Flag
Setup Size = 0x00004400
Magic signature found
Using boot protocol version 2.0c
Linux kernel version 3.14.17 (raop@hq1-ub-ecbld-373) #1 SMP Thu Jul 7 19:43:15 UTC 2016
```

The root account is now enabled. You can log in with the default *fibranne* password.

(XGS devices) Recover the root password

If you forgot the password for the VM root account, you can recover the default password.



Note

For a list of the currently supported XGS devices, see the "Supported Hardware" topic.

- Enter the **reboot** command.
- Select and enter the **ONIE** option.
- Select and enter the **ONIE: Rescue** option.
- Define the root login value for the root recover environment variable.

```
ONIE:/ # bootenv VM_Root_Recover RootPasswd
```

- Enter the **reboot** command.

```
ONIE:/ # reboot
ONIE:/ # discover: Rescue mode detected. No discover stopped.
Stopping: dropbear ssh daemon... done.
```

The root account is now enabled. You can log in with the default *fibranne* password.

- If the SLX-OS CLI is available, you can recover the root account by using the SLX-OS CLI **root enable** command.



Python Event-Management and Scripting

[Python under Extreme operating systems](#) on page 148

[Python scripts](#) on page 150

[Python event-management](#) on page 158

[Troubleshooting event-management](#) on page 160

[Event-management show commands](#) on page 160

Python under Extreme operating systems

The Python interpreter installed with supported Extreme operating systems enables you to access a Python shell or to launch Python scripts. You can also define event handlers that run such scripts automatically upon specified conditions.



Note

SLX-OS is among the Extreme operating systems that support Python.

Python overview

Python is a high-level scripting language that also supports object-oriented programming. If you have previous programming experience, you can quickly learn how to write useful, simple Python scripts.



Note

For Python resources, refer to <http://python.org>.

Working interactively in the Python shell

Use this procedure to access a Python shell, within which you can use Python commands that call and manipulate Extreme operating system commands.



Note

The Python shell is accessible only to admin-role users.
Python syntax is case-sensitive.

1. In privileged EXEC mode, enter **python** to access the Python shell.

```
device# python
```

The `device#` prompt changes to a Python prompt:

```
device# python
Python 3.5.2 (default, Apr 11 2019, 13:05:18)
[GCC 4.8.2] on linux
Type "help", "copyright", "credits" or "license" for more information.
```

2. To exit the Python shell and return to the Extreme operating system prompt, enter either:

- `exit()`
- `Ctrl-D`

3. To run a Extreme operating system command from within the Python shell, enter the `CLI ()` command.

```
>>> cmd_show_running_ve = CLI('show running-config interface ve')
!Command: show running-config interface ve
!Time: Mon Aug 22 16:53:13 2019
```

The statement entered above does two things:

- Runs the **show running-config interface ve** command and displays the result.
- Assigns that command to a Python variable named `cmd_show_running_ve`

4. To run a series of Extreme operating system commands from within the Python shell, separate the commands with `\n`.

```
>>> cmd_config_ve = CLI('configure \n interface ve 101-103')
!Command: configure
interface ve 101-103
!Time: Mon Aug 22 16:53:13 2019
```



Note

There is a difference between running a sequence of Extreme operating system CLI commands in the Python shell rather than in the standard Extreme operating system interface. Whereas in the standard interface the result of a command is persistent, in the Python shell each `CLI ()` statement is independent of any preceding ones.

In the following example, the lines beginning with **#** are added for explanation.

```
device# python
Python 3.5.2 (default, Apr 11 2019, 13:05:18)
[GCC 4.8.2] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> cmd_show_running_ve = CLI('show running-config interface ve')
!Command: show running-config interface ve
!Time: Mon Aug 22 16:53:13 2019

# No entries found.
```

```
# The SLX-OS show running-config interface ve command is run,  
# and that command is assigned to the Python variable cmd_show_running_ve.  
  
>>> cmd_config_ve = CLI('configure \n interface ve 101-103')  
# A series of three commands are run and assigned to the Python variable cmd_config_ve.  
!Command: configure  
  interface ve 101-103  
!Time: Mon Aug 22 16:53:13 2019  
  
>>> cmd_show_running_ve.rerun()  
# The rerun() function appended to cmd_show_running_ve gives the following output:  
!Command: show running-config interface ve  
!Time: Mon Aug 22 16:53:13 2019  
  
interface Ve 101  
  shutdown  
!  
interface Ve 102  
  shutdown  
!  
interface Ve 103  
  shutdown  
!  
!
```

Python scripts

Python scripts enable you to manipulate and launch Extreme operating system commands, taking advantage of the power and flexibility of Python. Such scripts also support event handling.

The topics in this section guide you through the process of writing and testing Python scripts, copying them to supported devices, and running them with the **python** command from the command line.

Guidelines for writing Python scripts

The general guidelines for writing Python scripts to run under SLX-OS are as follows:

- Although previous experience programming in Python is helpful, experience in other high-level languages is enough to get you started with simple Python scripts.
- The Python developer either needs experience with SLX-OS CLI or access to a resource with such experience.
- To help decide which editor or integrated development environment (IDE) to use, refer to <http://www.python.org>.
- Make sure that the appropriate version of the Python interpreter is installed on your development computer. For the current version of SLX-OS, install Python 3.5.2.
- Make sure that in the *Extreme SLX-OS Command Reference* you are familiar with the **python** and the **CLI ()** topics.
- The script must include `from CLI import CLI`. This enables the **CLI ()** command, by which Python can interact with SLX-OS.
- Write the Python script and save it, with a `.py` suffix. Valid filenames range from 4 through 32 characters (including the suffix). The first character must be alphabetic.

Guidelines for RASLOG event-handler scripts

The additional guidelines for writing RASLOG event-handler scripts are as follows:

- The script must include `import json`.
- If an event-handler detects the log messages specified in the triggers and the trigger-function conditions are met, the script is executed. As part of this execution, the script is passed a `--raslog-triggers` argument with JSON-formatted dictionary containing the relevant message identifiers (MSGID) as keys and the message text as values.
- If the trigger-function is OR, only a single MSGID and message text are present.
- If the trigger function is AND, all relevant MSGIDs and messages are passed inside the json data structure.



Note

For sample scripts, refer to [Python scripts and run-logs](#) on page 153.

Testing Python-script statements

While developing a Python script, you can test Extreme operating system calls by entering them in the device Python command shell. After the script is stable, you copy it to the device and then test it further by running it from the command line.

1. In privileged EXEC mode, enter the **python** command to access the Python shell.

```
device# python
Python 3.5.2 (default, Apr 11 2019, 13:05:18)
[GCC 4.8.2] on linux
Type "help", "copyright", "credits" or "license" for more information.
```

Note that the `device#` prompt changed to a `>>>` Python prompt:

2. Enter the script statements one at a time, verifying that they run as expected.

```
>>> cmd_config_ve = CLI('configure \n interface ve 101-103')
!Command: configure
  interface ve 101-103
!Time: Mon Aug 22 16:53:13 2019
```

3. Make corrections as needed.

Copying Python files to the device

After writing and testing a Python script file, use one of these topics to copy it to device flash memory.

Copying a file from a USB device

Use this topic to copy a file from a USB stick to an Extreme device.



Important

The only supported USB device for this task is the Extreme USB stick shipped with the device.

1. Copy the Python script file to the USB stick.
2. Insert the USB stick into the device USB port and enter the **usb on** command.

3. In privileged EXEC mode, enter the **copy** command to copy the Python file from the USB stick to the device flash memory.

```
device# copy usb://pythscript1.py flash://pythscript1.py
```

4. To display a list of files in the device flash memory, enter the **dir** command.

```
device# dir
total 32
drwxr-xr-x 2 251 1011 4096 Aug 26 08:44 .
drwxr-xr-x 3 251 1011 4096 Jul 20 07:50 ..
-rw-r--r-- 1 root root 1051 Mar 24 16:09 create_po.py
-rw-r--r-- 1 root sys 695 Aug 23 21:18 defaultconfig.cluster
-rw-r--r-- 1 root root 410 Aug 26 04:06 defaultconfig.standalone
-rw-r--r-- 1 root root 10042 Aug 4 00:01 ospfnsr1.cfg
-rw-r--r-- 1 root root 410 Aug 26 01:28 startup-config

16908197888 bytes total (10226114560 bytes free)
```

5. To display the contents of a Python file copied into the device flash memory, enter the **show file** command.

```
device# show file pythscript1.py
```

Downloading a file from a network

Use this topic to download a file from a network location to the Extreme device.

1. Make sure that the Python script file is uploaded to an accessible network location.
2. In privileged EXEC mode, enter the **copy** command to copy the Python file from the network location to the device flash memory.

```
device# copy ftp://MyUserID:MyPassword@10.10.10.10//pythscript1.py flash://
pythscript1.py
```

For other file-transfer options, refer to the *Extreme SLX-OS Command Reference* **copy** topic.

3. To display a list of files in the device flash memory, enter the **dir** command.

```
device# dir
total 32
drwxr-xr-x 2 251 1011 4096 Aug 26 08:44 .
drwxr-xr-x 3 251 1011 4096 Jul 20 07:50 ..
-rw-r--r-- 1 root root 1051 Mar 24 16:09 create_po.py
-rw-r--r-- 1 root sys 695 Aug 23 21:18 defaultconfig.cluster
-rw-r--r-- 1 root root 410 Aug 26 04:06 defaultconfig.standalone
-rw-r--r-- 1 root root 10042 Aug 4 00:01 ospfnsr1.cfg
-rw-r--r-- 1 root root 410 Aug 26 01:28 startup-config

16908197888 bytes total (10226114560 bytes free)
```

4. To display the contents of a Python file copied into the device flash memory, enter the **show file** command.

```
device# show file pythscript1.py
```


Running Python scripts from the command line

The facility to run Python scripts from the Extreme operating system command line enables you to execute complex and repetitious tasks with accuracy and efficiency. This facility also enables you to validate scripts intended for event management.



Caution

Make sure that you test Python scripts according to standard quality assurance practices before deploying them.

In privileged EXEC mode, enter the **python** command, specifying the Python script file that you want to run.

```
device# python create_po.py
```

After the script runs, the SLX-OS prompt displays.



Note

The create_po.py script is discussed in "Script for assigning interfaces to port channels (create_po.py)."

Python scripts and run-logs

This section contains sample SLX-OS Python scripts and run-logs.



Note

To access sample scripts and related resources, refer to <https://github.com/extremenetworks/ExtremeScripting>.

Script for assigning interfaces to port channels (create_po.py)

The create_po.py script is an example for automating common configuration tasks. This script runs the relevant **show running-config** commands before and after the following actions:

- VLAN configuration
- Port-channel configuration
- Adding interfaces to port channels



Note

Lines beginning with # are annotations.

```
#Required in all scripts for SLX-OS:
from CLI import CLI

slot = [0]
interfaces = [28, 29, 30, 31]
port_channel = 10
vlan_range = "101-105"

# Runs show running-config int vlan before the configuration, and assigns this SLX
# command to a Python variable named cmd_show_running_vlans.
cmd_show_running_vlans = CLI("show running-config vlan")

# Configures VLANs. {} is a placeholder for the format (arg1, arg2, ... argN) variables:
cmd_configure_vlans = CLI("config \n vlan {}".format(vlan_range))

# Reruns show running-config int vlan, following definition of new VLANs:
```

```

cmd_show_running_vlans.rerun()

# Configures port channel (vLAG):
cmd_show_running_port_channels = CLI("show running-config int po")
cmd_configure_port_channel = CLI("config \n int po {} \n switchport \n switchport mode
trunk \n switchport trunk allowed vlan add {} \n switchport trunk tag native-vlan ; no
shut".format(port_channel, vlan_range))

# Reruns show running-config int po, following configuration changes:
cmd_show_running_port_channels.rerun()

# Adds interfaces to port channel (vLAG)

for interface in interfaces:
    cmd_configure_interface = CLI("config \n int eth {}/{} \n channel-group {} mode
active type standard \n no shut".format(slot, interface, port_channel))
    cmd_show_running_int_tengig = CLI("show running-config int eth {}/{}".format (slot,
interface))

# Runs show running-config:
cmd_show_running = CLI("show running-config")

```

Run-log (*create_po.py*)

A log upon running *create_po.py* was as follows:

```

SLX# python create_po.py
Command: show running-config vlan
!Time: Fri Dec 16 18:35:41 2016

vlan 1
!
vlan dot1q tag native

!Command: config
vlan 101-105
!Time: Fri Dec 16 18:35:41 2016

!Command: show running-config vlan
!Time: Fri Dec 16 18:35:41 2016

vlan 1
!
vlan 101
!
vlan 102
!
vlan 103
!
vlan 104
!
vlan 105
!
vlan dot1q tag native

!Command: show running-config int po
!Time: Fri Dec 16 18:35:41 2016

% No entries found.

!Command: config
int po 10
switchport
switchport mode trunk

```

```

switchport trunk allowed vlan add 101-105
switchport trunk tag native-vlan ; no shut
!Time: Fri Dec 16 18:35:41 2016

!Command: show running-config int po
!Time: Fri Dec 16 18:35:42 2016

interface Port-channel 10
switchport
switchport mode trunk
switchport trunk allowed vlan add 101-105

```

Script illustrating the `.get_output` function (`get_output.py`)

The `.get_output` function returns—as a list—the output of the SLX-OS CLI commands assigned to a Python object.

Running this script displays the "Firmware name" line of the **show version** command.

```

#Required in all scripts for SLX:
from CLI import CLI

# Import the Python Regular Expressions (re) module:
import re

# Create Python objects:
slot_firmware = {}
cmd_show_ver = CLI("show ver", False)

# Using .get_output(), assign the result of show ver to a Python object named output:
output = cmd_show_ver.get_output()

for line in output:
    found = re.search(r'^(Firmware name:)\s+(\S+)$', line, re.M)
    if found:
        slot_firmware[found.group(1)] = found.group(2)

print("FIRMWARE:\n")
for key in slot_firmware:
    print("\t", key, "\t=> ", slot_firmware[key])

```

Run-log (`get_output.py`)

A log upon running `get_output.py` was as follows:

```

device# python get_output.py
FIRMWARE:

    Firmware name: 20.1.1_190404_0333

```

Event-handler script

This is a typical script launched when a specific RASLOG message is generated.

The `all_ports_down.py` script below was deployed—as a temporary workaround—to ensure that if a port-channel is shut down the members are also shut down.

**Note**

To access this script and related resources, refer to <https://github.com/extremenetworks/ExtremeScripting>.

```
#!/usr/local/python/3.5.2/bin/python3
import getopt
import json
import sys
import io
import re
import pdb
from CLI import CLI
int_down_raslog = 'NSM-1020'
int_up_raslog = 'NSM-1019'
shutdown_string = ''
match = None
raslog_triggers = {}
output = ''

# For argument processing:
options, remainder = getopt.gnu_getopt(sys.argv[1:], '', ['raslog-triggers='])

for opt, arg in options:
    if opt in ('--raslog-triggers'):
        print('--raslog-triggers: ', arg)
        output = io.StringIO(arg)
        raslog_triggers = json.load(output)

# Opening a file for logging:
f = open("poout.txt", "a")
f.write("testing:\n")

def po_members(po_num, shutdown_str):
    # Determines the physical members of the port-channel and executes
    # a shutdown/no shutdown command as requested for each physical interface:
    print("inside po_members", po_num, shutdown_str)
    f.write("inside po_members" + str(po_num) + str(shutdown_str))
    members = []
    # Executing the show port-channel CLI and matching the output:
    poCLI = "show port-channel " + str(po_num) + " | begin Link:"
    output = CLI(poCLI, do_print=False).get_output()
    print("output =\n" + str(output))
    f.write("output = \n" + str(output))

# Iterating over each interface found in the output, and executing the
# appropriate configuration commands:
for entry in output:
    print(entry)
    str1 = "".join(entry)
    str1 = str1.lstrip()
    print(str1.startswith("Link"))
    if str1.startswith("Link"):
        phy = str1.split("Link: ")[1].split("0x")[0]
        members.append(phy)
        command = "config term\nint " + phy + "\n" + str(shutdown_str)
```

```

        output = CLI(command, do_print=False).get_output()
        print("cli output = " + command + "\n" + str(output))
        f.write("cli output = " + command + "\n" + str(output))
        str1 = ""

    return members

# The following section uses the passed information from the event-handler
# and calls po_members with the correct information:
print('raslog_triggers:\n', str(raslog_triggers))
f.write("raslog_triggers:\n" + str(raslog_triggers) + '\n')
if int_down_raslog in raslog_triggers:
    shutdown_string = 'shutdown'
    match = re.search(r'interface Port-channel (\d+)',
                      raslog_triggers[int_down_raslog], re.IGNORECASE)
elif int_up_raslog in raslog_triggers:
    shutdown_string = 'no shutdown'
    match = re.search(r'interface Port-channel (\d+)',
                      raslog_triggers[int_up_raslog], re.IGNORECASE)

if match:
    po = match.group(1)
    print('Performing operation "' + shutdown_string
          + '" on members for Port-channel ' + po + '\n')
    f.write('Performing operation "' + shutdown_string
            + '" on members for Port-channel ' + po + '\n')
    members = po_members(po, shutdown_string)
    print('\tMembers on Port-channel ' + po + ': ' + str(members) + '\n')
    f.write('\tMembers on Port-channel ' + po + ': ' + str(members) + '\n')

f.close()

```

Test-run (Event-handler script)

The following command tests the `all_ports_down.py` script for an NSM-1020 RASLOG:

```
device# python all_ports_down.py --raslog-triggers {"NSM-1020":"interface Port-channel 10
is administratively down."}
```

The following command tests the `all_ports_down.py` script for an NSM-1019 RASLOG:

```
device# python all_ports_down.py --raslog-triggers {"NSM-1019":"interface Port-channel 10
is administratively up."}
```

To verify that the script with "NSM-1019" works correctly, enter the following commands:

```
device# show port-channel detail
LACP Aggregator: Po 10
Aggregator type: Standard
Actor System ID - 0x8000,78-a6-e1-45-95-14
Admin Key: 0010 - Oper Key 0010
Receive link count: 4 - Transmit link count: 4
Individual: 0 - Ready: 1
Partner System ID - 0x0001,00-24-38-8b-f1-00
Partner Oper Key 0102
Flag * indicates: Primary link in port-channel
Number of Ports: 4
Minimum links: 1
Member ports:
  Link: Eth 0/3 (0xC006000) sync: 1
  Link: Eth 0/4 (0xC008000) sync: 1 *
  Link: Eth 0/5 (0xC00A000) sync: 1
  Link: Eth 0/6 (0xC00C000) sync: 1

device# flex-cli show interface brief
Port   Link   Port-State   Dupl   Speed   Tag   MAC   Name

```

po10	Up	N/A	N/A	40000 Mbit	Yes	78a6.e145.9562	
po11	Up	N/A	N/A	30000 Mbit	Yes	78a6.e145.9563	
lb1	Down	N/A	N/A	N/A	N/A	N/A	N/A
0/1	Disabled	None	None	None	No	78a6.e145.9519	Ifml
0/2	Disabled	None	None	None	No	78a6.e145.951a	
0/3	Up	Forward	Full	10000 Mbit	No	78a6.e145.951b	
0/4	Up	Forward	Full	10000 Mbit	No	78a6.e145.951c	
0/5	Up	Forward	Full	10000 Mbit	No	78a6.e145.951d	
0/6	Up	Forward	Full	10000 Mbit	No	78a6.e145.951e	
0/7	Up	Forward	Full	10000 Mbit	No	78a6.e145.951f	

Python event-management

Python event management enables you to specify a Python script that runs automatically upon specified conditions.

You specify which RASlog message triggers the script.

Configuring an event-handler profile

Use this procedure to create an event-handler profile, define one or more triggers for it, and specify a Python script that runs upon one or more trigger events.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **event-handler** command.

```
device(config)# event-handler eventHandler1
```

3. For each trigger that you need for a profile, enter the **trigger** command.

```
device(config-event-handler-eventHandler1)# trigger 1 raslog LOG-1001
```

The trigger event is RASlog message #LOG-1001.

4. Enter the **action python-script** command to specify a Python script that runs when the event-handler is triggered.

```
device(config-event-handler-eventHandler1)# action python-script example.py
```

5. To add an event-handler-profile description, enter **description**, followed by the description text.

```
device(config-event-handler-eventHandler1)# description This is a sample description.
```



Caution

Make sure that you test Python scripts according to standard quality assurance practices before deploying them.

The following example includes all of the steps.

```
device# configure terminal
device(config)# event-handler eventHandler1
device(config-event-handler-eventHandler1)# trigger 1 raslog LOG-1001
device(config-event-handler-eventHandler1)# action python-script example.py
device(config-event-handler-eventHandler1)# description This is a sample description.
```

The following example defines a trigger that uses POSIX extended REGEX to search for a match within a specified RASlog message ID.

```
device# configure terminal
device(config-event-handler-eventHandler1)# event-handler eventHandler2
```

```
device(config-event-handler-eventHandler2)# trigger 1 raslog NSM-1003 pattern Interface
Ethernet 1/[1-9] is link down
```

RASlog message NSM-1003 includes "**interface** *interface-name* is link down", indicating that an interface is offline because the link is down. The REGEX searches within such a message for an interface from 1/1 through 1/9.

Activating an event-handler

Use this procedure to activate one or more event-handlers on the device. If a trigger specified in the event-handler profile occurs, a designated Python script runs.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **event-handler activate** command, specifying the event handler that you are activating.

```
device(config)# event-handler activate eventHandler1
```

3. To activate an additional event handler, enter the **event-handler activate** command, specifying the additional event handler.

```
device(config-activate-eventHandler1)# event-handler activate eventHandler2
```

4. To de-activate an event handler, enter the **no event-handler activate** command

```
device(config-activate-eventHandler2)# no event-handler activate eventHandler2
```

The following example includes all of the activation steps.

```
device# configure terminal
device(config)# event-handler activate eventHandler1
device(config-activate-eventHandler1)# event-handler activate eventHandler2
```

Configuring event-handler options

After you activate an event handler, you can configure various options. For example, you can specify if the event-handler script runs more than once and how multiple triggers are handled.

1. Activate an event handler, as described in [Configuring an event-handler profile](#) on page 158:

```
device# configure terminal
device(config)# event-handler activate eventHandler1
```

2. To specify a delay from when a trigger is received until execution of the event-handler action, enter the **delay** command.

```
device(config-activate-eventHandler1)# delay 60
```

The above example specifies a delay of 60 seconds.

3. To specify multiple iterations of the action when a trigger is received:

- a. Enter the **iterations** command.

- b. To specify an interval between iterations, enter the **interval** command.

```
device(config-event-handler-eventHandler1)# iterations 3
device(config-activate-eventHandler1)# interval 30
```

The above example sets the number of iterations to 3 and specifies an interval of 30 seconds between each iteration.

- To specify a maximum number of minutes to wait for an action script to complete execution, enter the **action-timeout** command.

```
device(config-activate-eventHandler1)# action-timeout 30
```

The example sets the timeout to 30 minutes.

- To limit action-recurrence upon multiple trigger-events, enter one of the following commands:

- trigger-mode only-once**—for the duration of a device configuration, the event-handler action is launched only once.

```
device(config-activate-eventHandler1)# trigger-mode only-once
```

- trigger-mode on-first-instance**—as long as the device is running, the event-handler action is launched only once. Following a device restart, the event-handler action can be triggered again.

```
device(config-activate-eventHandler1)# trigger-mode on-first-instance
```

- If multiple triggers are defined, to specify that the action run only if all of the triggers occur, enter the **trigger-function AND time-window** command.

```
device(config-activate-eventHandler1)# trigger-function AND time-window 120
```

The above example specifies that the action run only if all triggers occur within 120 seconds.

Troubleshooting event-management

Use these topics to troubleshoot issues that arise during implementation of Python event-management.

Aborting an event-handler action

If needed, abort a Python script launched by an event-handler action.

- In privileged EXEC mode, enter the **event-handler abort action** command.

```
device# event-handler abort action eh1
This operation will abort an event handler action that is currently running and may
leave the device in an inconsistent state. Do you want to continue? [y/n]:y
```

- To confirm aborting the action, type **y**.

```
Operation completed successfully.
```

The Python script launched by the event-handler action is aborted.

Event-management show commands

There are several show commands that display event-management information, listed here with descriptions.

Table 27: Event-management show commands in the *Command Reference*

Command	Description
show event-handler activations	Displays operational data of activated event-handlers.
show running-config event-handler	Displays details of event-handler profiles defined on the device. You can display the results by Python-script action or trigger ID.



Configuration Rollback

[Configuration rollback overview](#) on page 161

[Configuration rollback details](#) on page 163

[Configuration rollback considerations and limitations](#) on page 164

[Configuring rollback](#) on page 166

Configuration rollback overview

The configuration rollback feature provides the capability to take a checkpoint or snapshot of the current running configuration on the device and revert the current running configuration to a checkpoint configuration at a later stage, without the need to reboot the device.

This functionality can be used to revert to a previous configuration state, effectively rolling back any configuration changes that were made since that configuration checkpoint was created. Administrators can create multiple checkpoints to save different versions of the running configuration.

Supported topologies

The following IP Clos and non-Clos rack topologies are supported.

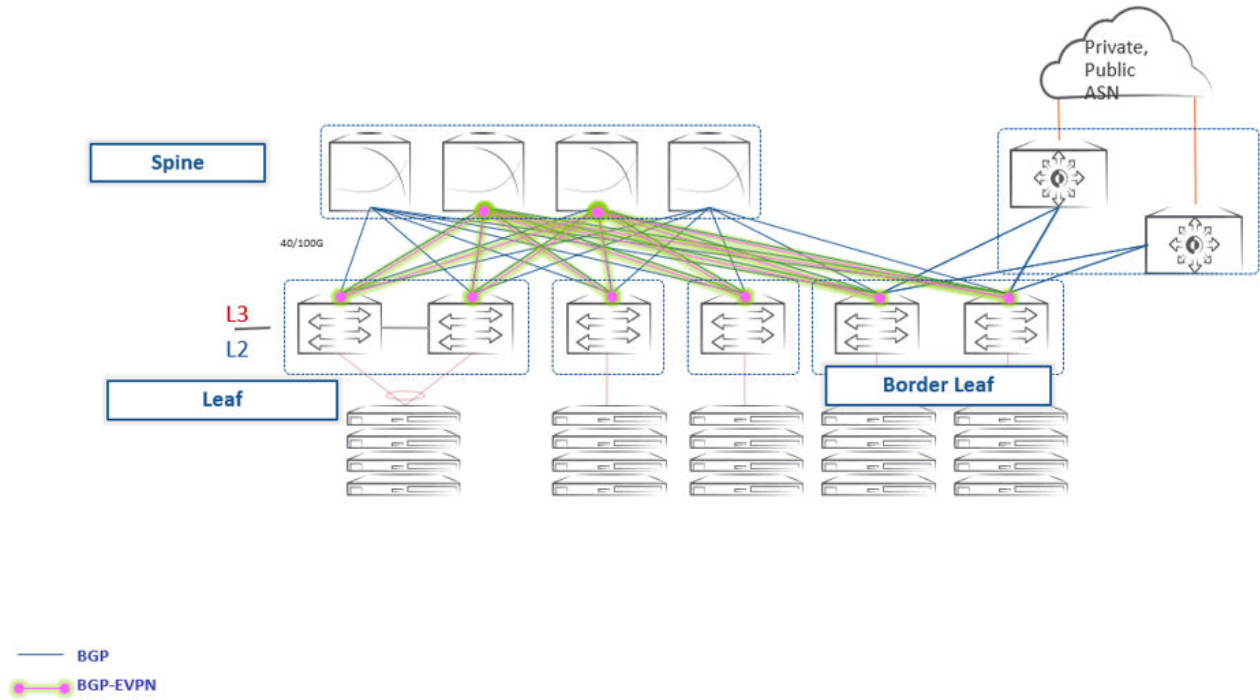


Figure 13: 3-tier IP Clos topology

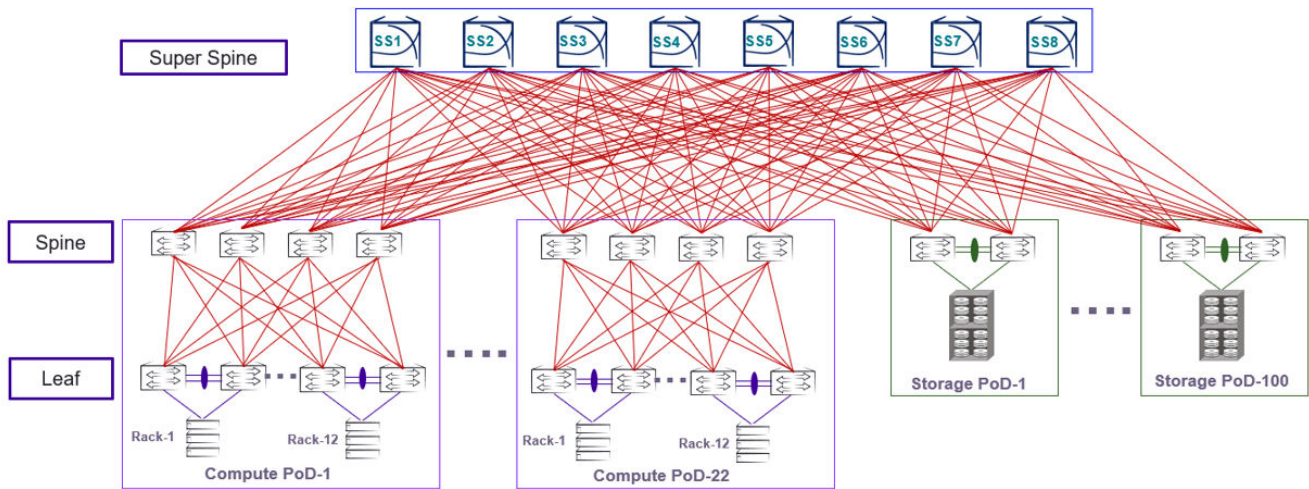


Figure 14: 5-tier IP Clos topology

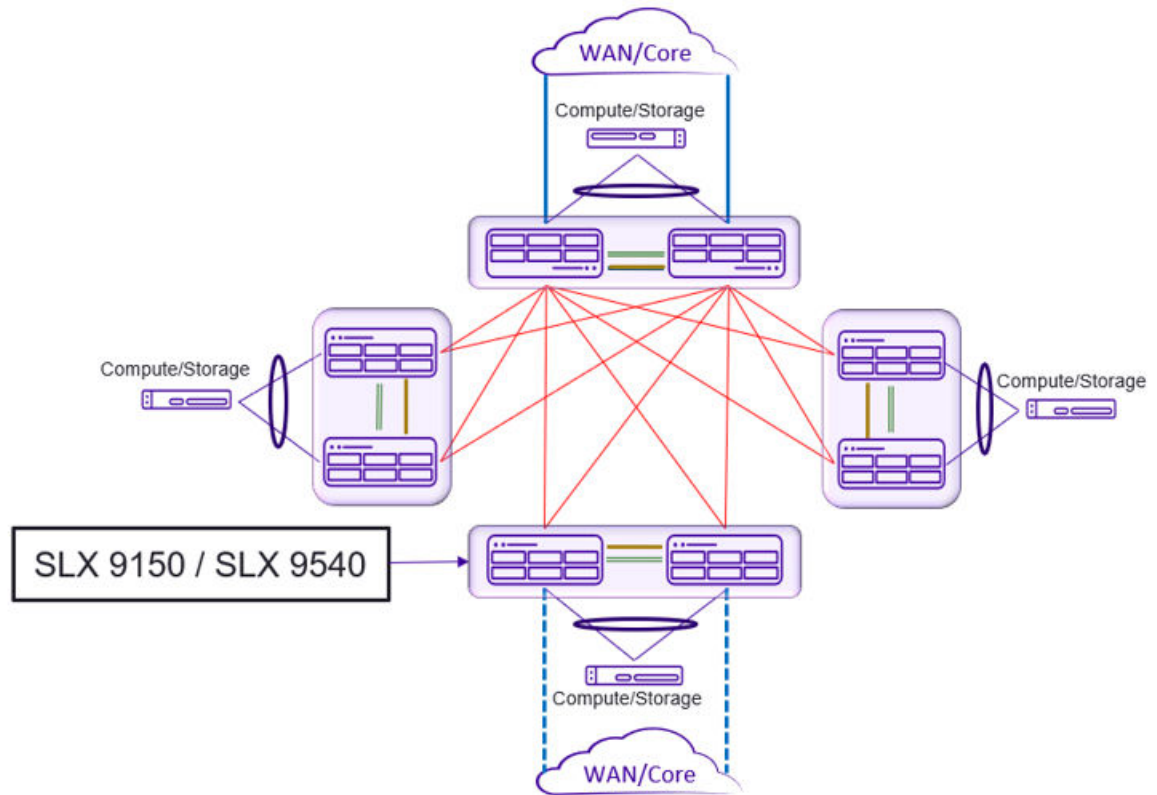


Figure 15: Non-Clos 4-rack topology

Configuration rollback details

This section provides additional details about the configuration rollback feature.

This feature is disabled by default. It is enabled by means of the **rollback enable** EXEC command. The execution of the **no rollback enable** command erases all the checkpoints and rollback related logs. Admin privileges are required to create checkpoints and perform rollback operations.

When rollback is disabled, other rollback commands result in the following error message:

```
Error%% This command is available when rollback is enabled.
```

The user creates configuration checkpoints (maximum of 10 are allowed) by executing the **rollback checkpoint** command in privileged EXEC mode. A checkpoint name is optional. If a name is not supplied, a checkpoint is created with a timestamp in *YYYYMMDD_HHMISS* format, for example, 20180511_234535.

As long as the checkpoint file is present on the device, the user can revert to a specified checkpoint file without having to reboot the device. However, certain configuration changes may require reboot, for example, hardware profile configurations. A rollback to a specific checkpoint restores the active configuration of the system to that of the checkpoint.

Two types of rollback are supported, by means of options to the **rollback apply checkpoint** command:

- **best-effort**: Implements a rollback and skips any errors (the default).
- **stop-at-first-failure**: Stops at the first error encountered.

A variety of show commands allow the user to see the details of failed configurations.



Note

MPLS and its allied configurations (VPLS, VLL, LDP) are not supported under Configuration Rollback.

Configuration rollback considerations and limitations

This section describes the rollback considerations and limitations of the SLX platform.

General

In a rollback operation, configuration diffs are generated between the running config and the checkpoint config. These can be seen by means of the **show rollback diff checkpoint** command. Configuration parameters that are changed are first removed, and then previous configuration parameters are reapplied.

The **show rollback patch checkpoint** command displays the patch file, which lists the sequence of CLI commands to be executed as part of a rollback.

Configurations from all other CLI/NETCONF/REST/RestConf and SNMP sessions are blocked when the rollback operation is in progress, with an error message as in the following CLI example.

```
device(conf-if-eth-0/1)# switchport
Rollback configuration is in progress. Please try again later.
device(conf-if-eth-0/1)#
```

Rollback and checkpoint operations are not permitted when the file/configuration replay operation is in process. An example error message is shown below.

```
device# rollback running-config checkpoint default
This operation will modify the running configuration of the system. Do you want to
continue? [Y/N]Y
%ERROR: Configuration rollback not allowed when file replay is in progress, try again
later
```

Only one rollback session is allowed. Subsequent attempts to roll back when there is an active rollback session are blocked, as in the following example.

```
device# rollback running-config checkpoint default
This operation will modify the running configuration of the system. Do you want to
continue? [Y/N]Y
%ERROR: There is another configuration rollback session in progress, try again later
```

A rollback operation is not permitted when cluster formation is in progress, as in the following example.

```
device# rollback running-config checkpoint default
This operation will modify the running configuration of the system. Do you want to
continue? [Y/N]Y
%ERROR: Cluster formation is in progress, try again later
```

All checkpoints and related artifacts (such as logs, history, and so on) are deleted from the system for the following conditions:

- When write-erase is issued
- When upgrade/downgrade is issued
- When **no rollback enable** is executed from global configuration mode.

When a firmware download is done with a full install, rollback will be disabled. When the device comes up with the new image, all the checkpoints will be lost.

Issues with specific configurations

In very rare cases, certain configuration commands cannot be removed from the running configuration without the device having to be reloaded. A configuration rollback operation that attempts to remove such a command could result in error messages indicating that these specific command lines have failed.

RAS considerations

The following table lists conditions and messages for Reliability, Availability, and Serviceability (RAS).

Table 28: RAS conditions and messages

Condition	Message
Rollback operation	Configuration Rollback to checkpoint <i><checkpoint-name></i> has started.
Rollback completion	Configuration Rollback to checkpoint <i><checkpoint-name></i> has been completed successfully.
	Configuration Rollback to checkpoint <i><checkpoint-name></i> has been aborted.
	Configuration Rollback to checkpoint <i><checkpoint-name></i> has failed. Please use <code>show rollback log [errors]</code> to see the reasons for the failure.
Checkpoint creation	Checkpoint <i><checkpoint-name></i> is created by <i><user></i> .
Checkpoint deletion	Checkpoint <i><checkpoint-name></i> is deleted by <i><user></i> .

Intrusive scenarios

The following are among the intrusive scenarios that can occur in moves from a running configuration to a checkpoint configuration.

- The running configuration has `config switchport trunk allowed vlan all` and the checkpoint has the range-based switchport trunk configuration `switchport trunk allowed vlan add <vlan-range>`. Such configurations result in traffic disruption while reverting to a checkpoint configuration.
- When network telemetry is activated in the running configuration and a checkpoint configuration has modifications for the telemetry server related configuration, the telemetry server must be deactivated and the changes applied.
- Certain feature configurations such as HTTPS and, telemetry are dependent on some exec mode commands for related artifacts like crypto-certificates. The rollback feature is unable to determine any discrepancies in such dependencies.

Performance considerations

A rollback operation involves retrieving an existing running configuration, computing the differences between that and a checkpoint configuration, and replaying the file of the diff that is generated. Where network scales are such that there are huge differences between the running configuration and checkpoint configuration, it can take several minutes to complete the rollback.

Configuring rollback

The following examples illustrate how to create a checkpoint file, view the rollback/diff patch, revert to a user-defined checkpoint, and view the status of the operation.

Enabling or disabling rollback

Use the **enable rollback** and **no enable rollback** commands to enable or disable rollback, respectively.

```
device(config)# rollback enable

device(config)#no rollback enable
%%WARN: All checkpoints and rollback logs will be cleared!
Do you want to continue? [y/n]:
```

Creating a default configuration checkpoint

Use the **rollback checkpoint** command to create a default configuration checkpoint.

```
device# rollback checkpoint default_config_checkpoint description "Default Config"
Checkpoint default_config_checkpoint creation request by user: admin
Checkpoint default_config_checkpoint creation completed successfully.
```

Viewing checkpoint details

Use the **show rollback checkpoint** command to view checkpoint details. The **summary** option is used here.

```
device# show rollback checkpoint summary
User checkpoint summary
-----
1) default_config:
Created by "admin"
Created at Tue Jun 12 14:19:49 2018
Size is 4880 bytes
Description: "Default Config"

2) vlan_config:
Created by "admin"
Created at Tue Jun 13 14:19:49 2018
Size is 4872 bytes
Description: "Vlan Config"
```

Modify the running configuration

Depending on the requirements, the running configuration can be modified as needed, as shown in the vlan configuration below.

```
device# conf t
Entering configuration mode terminal
device(config)# vlan 100
device(config-vlan-100)# name "VLAN 100"
device(config-vlan-100)# exit
device(config)# vlan 200
device(config-vlan-200)# name "VLAN 200"
device(config-vlan-200)# exit
device(config)# no snmp-server enable trap
device(config)# end
```

Viewing the diff between a checkpoint and the running configuration

Use the **show rollback diff checkpoint** command to view a diff between a checkpoint and a running configuration. Entries removed are indicated by "-" and entries added are indicated by "+".

```
device# show rollback diff checkpoint default_config_checkpoint
!
-snmpp-server enable trap
+!
+vlan 100
+ name VLAN 100
+!
+vlan 200
+ name VLAN 200
```

The following shows a diff between two checkpoints.

```
device# show rollback diff checkpoint default checkpoint switchport
interface Ethernet 1/10
- switchport port-security shutdown-time 10
- switchport trunk native-vlan 2
+ switchport port-security max 100
```

Viewing the patch between a checkpoint and the running configuration

Use the **show rollback patch checkpoint** command to view a patch between a checkpoint and the running configuration.

```
device# show rollback patch checkpoint default_config_checkpoint
!  
no vlan 200  
no vlan 100  
!  
snmp-server enable trap  
!
```

Executing rollback

Use the **rollback apply checkpoint** command to execute a rollback. To rollback the configuration to a specific, saved checkpoint, the rollback configuration name for that checkpoint must be used. Please note that any changes made to the configuration after the checkpoint was taken will be removed when the rollback is performed.

```
device# rollback apply checkpoint default_config_checkpoint  
This operation will modify the running configuration of the system. Do you want to  
continue? [Y/N]y  
% Warning: Configuration Rollback is in-progress.  
Please do not abort an ongoing session as it can leave the system with an inconsistent  
configuration.  
.....  
Rollback completed successfully.
```

Verifying that the rollback diff is empty

Use the **show rollback diff checkpoint** command to verify that the diff is empty.

```
device# show rollback diff checkpoint default_config_checkpoint  
device#
```

Viewing rollback status

Use the **show rollback status** command to view rollback status.

```
device# show rollback status  
Operation                : Rollback To Checkpoint  
Checkpoint Name          : default_config_checkpoint  
Rollback done By         : admin  
Rollback Mode            : Best Effort  
Start Time               : Tue Jun 12 14:27:04 2018  
End Time                 : Thu Jan 12 14:27:31 2018  
Time Taken               : 27 seconds  
Status                   : Success  
device#
```

Viewing the rollback log

Use the **show rollback log** command to view the rollback log.

```
device# show rollback log  
vlan 6  
no description
```



```

no suppress-arp
no ipv6 mld snooping startup-query-interval
no ip igmp snooping startup-query-interval
no suppress-nd
no router-interface Ve
!
exit
vlan 3
no description
no suppress-arp
no ipv6 mld snooping startup-query-interval
no ip igmp snooping startup-query-interval
no suppress-nd
no router-interface Ve
!
exit
vlan 2
no description
no suppress-arp
no ipv6 mld snooping startup-query-interval
no ip igmp snooping startup-query-interval
no suppress-nd
no router-interface Ve
!
exit
vlan 1
no ipv6 mld snooping startup-query-interval

no ip igmp snooping startup-query-interval
!
exit
no ip access-list standard acl1
no ip mtu
no ipv6 mtu
no mtu
no vrf CFD1722809
event-handler ev1
  action python-script show_interface.py
Error: flash://show_interface.py script for event handler ev1 could not be found or read.
router bgp
  neighbor 1.1.1.1 remote-as 13
Warning: Reset the neighbor session

```

Viewing rollback log errors

Use the **show rollback log errors** command to view log errors.

```

device# show rollback log errors
no bridge-domain 1 p2mp
%Error: One or more specified vlan(s) has a VE configured.
port 50055
% Error: Cannot update/delete port and transport configurations once the Telemetry server
is activated.

```

Viewing current rollback status

Use the **show rollback status current** command to view current status.

```

device# show rollback status current
Operation                : Rollback To Checkpoint
Checkpoint Name          : vlan-config
Rollback done By         : admin

```

```
Rollback Mode      : best-effort
Start Time         : Thu Apr  5 09:32:24 2018
Status             : In-Progress
```

Viewing rollback status history

Use the **show rollback status history** command to view status history.

```
device# show rollback status history
Operation          : Rollback To Checkpoint
Checkpoint Name    : vlan-config
Rollback done By   : admin
Rollback Mode      : best-effort
Start Time         : Thu Apr  5 09:32:24 2018
End Time          : Thu Apr  5 09:32:57 2018
Time Taken For Rollback : 33 seconds
Status            : Success

Operation          : Rollback To Checkpoint
Checkpoint Name    : bgp-config
Rollback done By   : admin
Rollback Mode      : best-effort
Start Time         : Thu Apr  5 07:32:24 2018
End Time          : Thu Apr  5 07:32:57 2018
Time Taken For Rollback : 38 seconds
Status            : Success
```