



Extreme SLX-OS Management Configuration Guide, 20.8.1

Supporting ExtremeRouting and ExtremeSwitching
SLX 9740, SLX 9640, SLX 9540, SLX 9250, SLX 9150,
Extreme 8820, Extreme 8720, and Extreme 8520

9041023-00 Rev AA
April 2026



Copyright © 2026 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, see: <https://www.extremenetworks.com/about-extreme-networks/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: <https://www.extremenetworks.com/support/policies/open-source-declaration/>

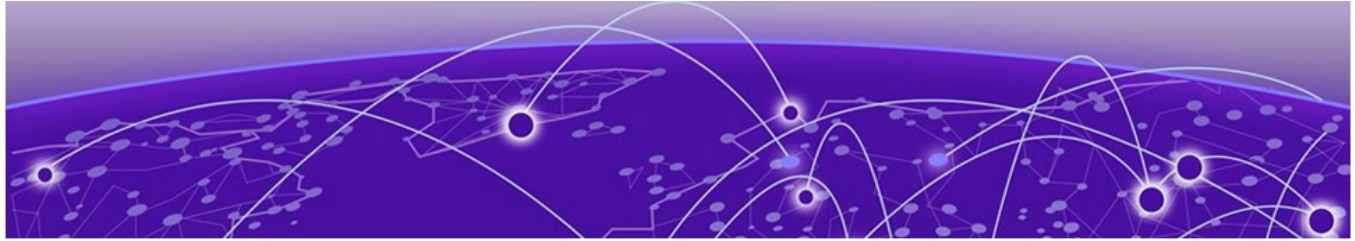


Table of Contents

Preface.....	9
Text Conventions.....	9
Documentation and Training.....	10
Open Source Declarations.....	11
Training.....	11
Help and Support.....	11
Subscribe to Product Announcements.....	12
Send Feedback.....	12
About This Document.....	13
What's New in this Document	13
Supported Hardware.....	13
Regarding Ethernet interfaces and chassis devices.....	14
Configuration Fundamentals.....	15
Configuration Files.....	15
Default Configuration Files.....	16
Startup Configuration Files.....	16
Running Configuration Files.....	16
Auto-Persistence of Configuration Data.....	16
Displaying configurations.....	18
Backing up a running configuration.....	19
Backing up configurations.....	19
Configuration restoration.....	20
Tracking Configuration Changes.....	21
Managing flash files.....	22
Rebooting the device.....	24
Copying Support Save Files	24
Session connection.....	26
Telnet.....	27
SSH.....	29
Configuring the terminal session parameters.....	34
Configuring banners on SLX-OS	35
Ethernet management interfaces.....	36
Configuring a static ethernet address.....	36
Displaying the management interface.....	37
Redundant Management Interface.....	37
Force 1G connection speed on physical interfaces.....	41
Configuring an IPv6 address on the SLX platform.....	43
Port management.....	44
SLX 100G ports.....	45
Configuring breakout mode.....	45

10G/1G auto negotiation and auto detection mode.....	47
Port flap dampening.....	48
Port transition hold timer.....	48
Link fault signaling.....	50
Interface Ethernet ports.....	51
Displaying device interfaces.....	51
Interface reload delay to prevent traffic black-holing in vLAG.....	52
Scenario 1.....	52
Scenario 2.....	53
Configuration examples.....	54
Chassis and host names.....	55
Customizing chassis and host names.....	55
System clock.....	56
Setting the clock.....	57
Management VRFs.....	57
VRF reachability.....	58
Heartbeat between SLX and EFA	61
When enabled for the first time.....	61
When heartbeat messages are not received.....	61
After a reboot	61
Transitioning between Management Operational State - Down and Up states.	61
Heartbeat Setting	62
Zero Touch Provisioning.....	63
Routing for ZTP.....	65
ZTP over HTTPS	65
Using ZTP.....	65
ZTP configuration.....	66
Example of ZTP in a two-node topology	72
MAC address aging.....	76
TCAM application-resource monitoring.....	77
TCAM library-resource monitoring.....	78
Hardware profiles.....	79
TCAM profiles.....	79
TCAM sharing.....	82
Counter profiles.....	83
FIB compression.....	84
Border profiles for Internet peering.....	86
Hardware profile show commands	88
Enter Maintenance Mode Before Performing Device Maintenance.....	88
Rebooting into Maintenance Mode	89
Support for OpenConfig Telemetry	90
Introduction.....	90
OpenConfig-Interface YANG Module.....	90
OpenConfig-BGP Yang Module	91
OpenConfig-Platform Yang Module	91
Enabling OpenConfig Telemetry Support	92
Securing OpenConfig Telemetry Connections	93
Importing gNMI Server Private Key and Server Certificate	93
Importing gNMI Client CA Root Certificate	94

Static Prefix Independent Convergence.....	94
Static Prefix Independent Convergence	94
SLX-OS and Linux Shell Interoperability.....	97
Overview	97
Limitations	97
Launching Linux shell from SLX-OS	98
Executing Linux shell commands from SLX-OS.....	99
Executing scripts from SLX-OS.....	100
Downloading a script to the SLX-OS device.....	100
Creating scripts in the Linux shell.....	100
Running scripts from the SLX-OS CLI.....	101
Accessing the Linux shell from SLX-OS.....	101
Executing SLX-OS commands from the Linux shell.....	101
Escalating Linux permissions to root.....	102
Saving and appending show command output to a file.....	103
Logs of Linux shell activities.....	103
Linux shell user entry and exit logs.....	103
Linux shell command execution logs.....	104
Configuring remote logging of Linux shell activities.....	105
Guest OS for TPVM.....	106
VM Access Management.....	106
Extreme SLX-OS VM Access Management.....	108
Insight Interface and TPVM.....	111
Insight interface port-channel.....	112
TPVM on the SLX 9150 series.....	113
Configuring the Insight Interface for the SLX 9150/9250.....	113
Insight interface.....	115
Inbound ACL-based mirroring.....	118
Insight interface traffic management and QoS.....	120
Configuring QoS egress scheduling.....	122
Troubleshooting port-mirroring.....	124
TPVM.....	127
Supported third-party applications, packages, and hardware.....	127
TPVM Management.....	130
Docker containers.....	143
Linux containers.....	145
Utilities installation and management.....	146
Assigning a static IP address on the TPVM Linux OS.....	148
Update TPVM Gateway.....	150
IPv6 Support for TPVM.....	150
TPVM Configuration Persistence	150
How it Works.....	151
Deploy with TPVM Configuration Persistence.....	154
Managing Snapshots.....	156
Upgrading TPVM	158
TPVM Migration	159
Remove TPVM on Copying Default Config as Startup Config	160
Uninstalling TPVM completely	160

Restoring Configuration from Backup File	161
Measured Boot.....	162
Measured Boot.....	162
Supported Platforms.....	162
Verifying Measured Boot	162
Remote Attestation.....	164
Remote Attestation	164
Keylime Components.....	165
Configure Remote Attestation	165
Network Time Protocol (NTP).....	168
Network Time Protocol overview.....	168
Date and time settings.....	168
Time zone settings.....	168
Network Time Protocol Server Overview.....	169
Network Time Protocol Client Overview.....	169
Network Time Protocol Associations.....	170
Network Time Protocol Authentication.....	171
Configuring NTP.....	172
Authenticating an NTP server.....	173
Displaying the active NTP server.....	174
NTP server status when an NTP server is not configured.....	174
NTP server status when an NTP server is configured.....	174
SNMP.....	175
SNMP overview.....	175
Basic SNMP operation.....	176
SNMP community strings.....	177
SNMP groups.....	177
SNMP users.....	178
SNMP views.....	178
SNMP server hosts.....	178
SNMP Engine ID.....	178
Multiple SNMP server context to VRF mapping.....	180
SNMP source interface.....	180
Configuring SNMPv2.....	181
Configuring SNMPv3.....	182
Configuring an SNMP server context to a VRF.....	183
Offline SNMP ifIndex generation tool.....	184
Generating ifIndexes for various interfaces.....	185
Configuration examples for generating ifIndexes offline.....	185
LLDP.....	187
LLDP overview.....	187
Layer 2 topology mapping.....	187
LLDP configuration guidelines and restrictions.....	189
Configuring and managing LLDP.....	190
Understanding the default LLDP.....	190
Disabling LLDP globally.....	190
Configuring LLDP global parameters.....	191

Configuring LLDP profiles.....	192
Configuring an LLDP profile to an interface.....	193
Displaying LLDP information.....	194
Clearing LLDP-related information.....	196
Account and Password Recovery.....	197
Recover the admin password from the root account.....	197
Root account and password recovery.....	197
Recover the root login account.....	198
Recover the root password when shell prompt is available.....	199
Recover the root password when ONIE is available.....	200
Force Password Change At First Login	200
Force Password Age Out.....	200
Configure an account to disable automatically.....	201
Configure an account with inactivity warning	202
Changing default password for the system default accounts.....	202
Python Event-Management and Scripting.....	203
Python under Extreme operating systems	203
Python overview	203
Launching Python interface from SLX-OS	203
Working interactively in the Python shell	205
Python scripts	207
Guidelines for writing Python scripts	207
Testing Python-script statements	208
Copying Python files to the device.....	208
Running Python scripts from the command line.....	209
Python scripts and run-logs	210
Python event-management	214
Configuring an event-handler profile	214
Activating an event-handler	215
Configuring event-handler options	216
Troubleshooting event-management.....	217
Aborting an event-handler action.....	217
Event-management show commands	217
Configuration Rollback.....	218
Configuration rollback overview.....	218
Supported topologies.....	218
Configuration rollback details.....	220
Configuration rollback considerations and limitations.....	221
General.....	221
Issues with specific configurations.....	222
RAS considerations.....	222
Intrusive scenarios.....	223
Performance considerations.....	223
Configuring rollback.....	223
Enabling or disabling rollback.....	223
Creating a default configuration checkpoint.....	224
Viewing checkpoint details.....	224
Modify the running configuration.....	224

Viewing the diff between a checkpoint and the running configuration.....	224
Viewing the patch between a checkpoint and the running configuration.....	225
Executing rollback.....	225
Verifying that the rollback diff is empty.....	225
Viewing rollback status.....	225
Viewing the rollback log.....	226
Viewing rollback log errors.....	226
Viewing current rollback status.....	227
Viewing rollback status history.....	227
BMC Configuration.....	228
Increase BMC Security	228
Intelligent Platform Management Interface	228
Baseboard Management Controller	229
Securing BMC	229
Change BMC User Password	229
Enable the BMC Management Interface	231
Configure BMC Management Interface IP Address.....	231
Managing BMC Management Interface	233
Reset BMC Configuration to Factory Defaults	234



Preface

Read the following topics to learn about:

- The meanings of text formats used in this document.
- Where you can find additional information and help.
- How to reach us with questions and comments.

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 Extreme Networks switches, the product is referred to as *the switch*.

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
	Caution	Risk of personal injury, system damage, or loss of data
	Warning	Risk of severe personal injury

Table 2: Text

Convention	Description
screen displays	This typeface indicates command syntax, or represents information as it is displayed 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!	New information. In a PDF, this is searchable text.

Table 3: Command syntax

Convention	Description
bold text	Bold text indicates command names, keywords, and command options.
<i>italic</i> text	Italic text indicates variable content.
[]	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[member...]</i> .
\	In command examples, the backslash indicates a “soft” line break. When 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 and Software Compatibility](#) for Extreme Networks products

[Extreme Optics Compatibility](#)

[Other Resources](#) such as articles, white papers, and case studies

Open Source Declarations

Some software files have been licensed under certain open source licenses. Information is available on the [Open Source Declaration](#) page.

Training

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For details, visit the [Extreme Networks Training](#) page.

Help and Support

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 2800. 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 Product Announcements

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

1. Go to [The Hub](#).
2. In the list of categories, expand the **Product Announcements** list.
3. Select a product for which you would like to receive notifications.
4. Select **Subscribe**.
5. To select additional products, return to the **Product Announcements** list and repeat steps 3 and 4.

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

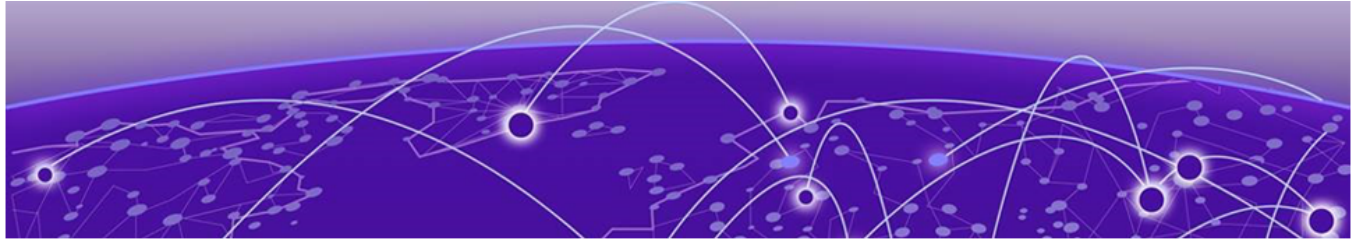
Send Feedback

The User Enablement team at Extreme Networks has made every effort to ensure that this document is accurate, complete, and easy to use. We strive to improve our documentation to help you in your work, 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.
- Broken links or usability issues.

To send feedback, email us at Product-Documentation@extremenetworks.com.

Provide as much detail as possible including the publication title, topic heading, and page number (if applicable), along with your comments and suggestions for improvement.



About This Document

[What's New in this Document](#) on page 13

[Supported Hardware](#) on page 13

[Regarding Ethernet interfaces and chassis devices](#) on page 14

What's New in this Document

This document is released with the SLX-OS 20.8.1 software release. The following changes were made to this document.

Feature	Description	Described in
SLX-OS and Linux Shell Interoperability	Added a topic that describes how to launch the Linux shell from within SLX-OS. This topic also discusses the <i>Disclaimer</i> that is shown on launching this shell interface.	Launching Linux shell from SLX-OS on page 98
Python Event Management and Scripting	Added a topic that describes how to launch the Python interactive interface from within SLX-OS. This topic also discusses the <i>Disclaimer</i> that is shown on launching this interface.	Launching Python interface from SLX-OS on page 203
Guest OS for TPVM	Added that the TPVM Debian file will be removed from the device post successful deployment or upgrade.	TPVM Package Information on page 132

For additional information, refer to the *Extreme SLX-OS Release Notes* for this version.

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 20.8.1 supports the following hardware platforms.

- Extreme 8820
- Extreme 8720
- Extreme 8520
- ExtremeSwitching SLX 9540

- ExtremeSwitching SLX 9250
- ExtremeSwitching SLX 9150
- ExtremeRouting SLX 9740
- ExtremeRouting SLX 9640

**Note**

All configurations and software features that are applicable to SLX 9150 and SLX 9250 devices are also applicable for the Extreme 8520 and Extreme 8720 devices respectively.

All configurations and software features that are applicable to SLX 9740 devices are also applicable for the Extreme 8820 devices.

The "Measured Boot with Remote Attestation" feature is only applicable to the Extreme 8520, Extreme 8720, and Extreme 8820 devices. It is not supported on the SLX 9150 and SLX 9250 devices.

**Note**

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

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.

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 15
[Session connection](#) on page 26
[Ethernet management interfaces](#) on page 36
[Configuring an IPv6 address on the SLX platform](#) on page 43
[Port management](#) on page 44
[Interface Ethernet ports](#) on page 51
[Interface reload delay to prevent traffic black-holing in vLAG](#) on page 52
[Chassis and host names](#) on page 55
[System clock](#) on page 56
[Management VRFs](#) on page 57
[Heartbeat between SLX and EFA](#) on page 61
[Zero Touch Provisioning](#) on page 63
[MAC address aging](#) on page 76
[TCAM application-resource monitoring](#) on page 77
[Hardware profiles](#) on page 79
[Enter Maintenance Mode Before Performing Device Maintenance](#) on page 88
[Rebooting into Maintenance Mode](#) on page 89
[Support for OpenConfig Telemetry](#) on page 90
[Static Prefix Independent Convergence](#) on page 94

Configuration Files

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

The startup configuration resides in the `/var/config/vcs/scripts` directory.

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

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.

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 are made to the running configuration.

- 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.

Auto-Persistence of Configuration Data

All configuration changes performed on the device are automatically persisted. You do not need to run the **copy running-config startup-config** command to persist these changes.

Configurations are saved to a persistent configuration data store. After a device reboots, or, is brought up, configurations are restored from this data store. If this data store becomes unusable for any reason, configurations are then replayed from the startup-config file.

Initially, the startup-config file has the *factory-default* configuration. When the **copy running-config startup-config** command is executed, the configurations in the running-config data store is copied to the startup-config data store. The same configuration changes are also updated to the startup-config file.

Controlling Configuration Replay During Device Boot

By default, configurations are restored from the startup data store when the device boots up after a reboot. To ensure that the device uses the startup-config file instead of the startup-config data store, execute one of the following commands:

- **copy default-config startup-config**
- **Copy <tftp:/scp:/ftp:/usb:/flash:>/<file> startup-config**

followed by the device reboot.

Before Upgrading or Downgrading

Before upgrading or downgrading the device's firmware, it is recommended that you execute **copy running-config startup-config** command. This will ensure that the startup-config data store and startup-config file are updated with the changes stored in the running-config data store.

Viewing the Various Configurations

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

- Displaying the Default Configuration:

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

```
(device)# show file defaultconfig.standalone
```

- Displaying the Startup Configuration

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

```
(device)# show startup-config
```

- Displaying the Running Configuration

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

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



Note

The `show startup-database` command is deprecated.

Examples

The following example displays the status of auto persistence in a SLX device.

```
SLX(config) # show auto-persistence status
              Auto persistence: Enabled
              SLX(config) #
```

The following examples illustrate how to enable and disable the auto persistence on a SLX device. Auto persistence is enabled by default. When you use the `auto-persistence disable` command, you disable this feature.

- To enable the auto persistence when it is disabled,

```
SLX(config)# no auto-persistence disable
```

- To disable the auto persistence when it is enabled,

```
SLX(config)# auto-persistence disable
```

To copy the running configuration to the startup configuration manually, use the `SLX(config)# copy running-config startup-config` command.

```
SLX (config)# copy running-config startup-config

                                This operation will back up the current configuration. Do you want to
continue? [y/n]:y
                                Running-config was committed to startup-config successfully.
```

To view the contents of the startup-configuration or the running-configuration data stores, use the `show startup-database` or the `running-database` commands.

```
SLX (config)# show startup-database VLAN
vlan 1
!
vlan 2
!
vlan 3
!
vlan 4
!
vlan 5
!
vlan dot1q tag native
SLX(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
```



Warning

It is recommended that you use the interactive way of providing password when you copy a configuration file to an external host using FTP/SCP. Providing the password in the URL will add the complete string including the password to the command history in plain text. The password will be exposed when the **show history** command is executed.

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 startup 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
```

Tracking Configuration Changes

SLX devices can be configured from either their console or from applications, such as EFA, that enable remote configuration. These applications need to keep their stored configurations synchronized with the configuration currently applied on the SLX device and use CLI commands, NETCONF, REST commands to achieve this synchronization.

SLX tracks changes made to its configuration through SSH/telnet/NETCONF by using a timestamp and counter that is incremented for every change that is made to the configuration. The timestamp keeps track of the exact time when the last configuration change was made.

Applications can use this information to determine if their stored configuration is out-of-sync with SLX's current configuration and needs to be updated. If the stored configuration needs to be updated, applications will continue to use existing methods of synchronization to update their configurations.

This feature is provided to enable applications to quickly detect differences in configuration and perform reconciliation only when required.



Note

This support is not available for configuration changes performed through REST/RESTCONF interface.

By default, tracking of configuration changes is enabled globally for the SLX device and cannot be switched off globally. However, if required, applications can temporarily

switch off configuration change detection for the current session. Any changes made in the session, where configuration change tracking is disabled, is not tracked by SLX.

**Important**

Timestamp values are based on the SLX device's clock and is subject to clock changes.

**Note**

The Configuration Drift Tracking counter value persists across reboots.

Turning off tracking of configuration changes

An application that does not want its configuration changes tracked by SLX, should turn off Configuration Drift Tracking in that session immediately after starting the session.

To turn off Configuration Drift Tracking for a session, execute the following command in CLI mode:

From the command prompt, run the **config-drift-track off** command.

```
SLX# config-drift-track off
SLX#
```

Configuration Drift Tracking is turned off for the current session.

**Note**

Configuration Drift Tracking will be enabled for a session until it gets turned off explicitly for that session.

**Note**

Any attempt to turn off Configuration Drift Tracking for a session where it is already turned off, gets ignored by SLX. Also, application need not turn off drift tracking for a read-only session.

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
```

Copying Support Save Files

When Support Save files are copied using the **copy support** command, they are all saved under a single directory on the remote server. The directory name is derived by combining the hostname and serial number of the SLX device and the timestamp of when the **copy support** command was executed.

A new directory with the format `<HOSTNAME>_<SERIAL-NO>_<MMDD_HHMM>` is automatically created on the remote server or USB device. The location of this sub-directory will depend on the **copy** command parameters.

Destination is a USB Drive

When the destination for saving the support save files is a USB device, the support save directory is always created in the directory `/usb/usbstorage/slxos/slxos/support/`.

For example, if the SLX host details are as under:

```
Hostname: SLX-MY-HOST
Serial-Number: TH000001Q-00001
Time Stamp: 23-OCT-2021 02:25:36 PM
```

Then the Support Save files are stored in the newly created directory *SLX-MY-HOST_TH000001Q-00001_1023_1425*.

```
/usb/usbstorage/slxos/slxos/support/SLX-MY-HOST_TH000001Q-00001_1023_1425
```

Destination is a remote FTP server

When using a remote FTP server to transfer your Support Save files, the destination folder must be passed with the **copy support** command. You need not create the destination directory. The **copy support** command will create the remote directory structure before creating the Support Save sub-directory within it. It will then copy the Support Save files into this newly created sub-directory.

For example, if the SLX host details are as under:

```
Hostname: SLX-MY-HOST
Serial-Number: TH000001Q-00001
Time Stamp: 23-OCT-2021 02:25:36 PM
```

and the FTP server details are as under:

```
Hostname: 10.25.37.42
User-name: ftp-user
Password: password
Destination Directory: /ftp-dir/support-saves/
```

Then the Support Save files are stored in the newly created directory *SLX-MY-HOST_TH000001Q-00001_1023_1425*. If the destination directory */ftp-dir/support-save* is not available, then it will be created automatically and the directory *SLX-MY-HOST_TH000001Q-00001_1023_1425* created within it. All the Support Save files are then copied to the newly created directory. The path under which you will find the Support Save files will be:

```
/ftp-dir/support-saves/SLX-MY-HOST_TH000001Q-00001_1023_1425
```

Destination is a remote server and SCP is used to transfer the files.

When using SCP to transfer the Support Save files to a remote server, the destination folder must be passed with the **copy support** command. The Support Save files will be created in the directory named *SLX-MY-HOST_TH000001Q-00001_1023_1425* within the destination directory. If the destination directory is not present on the remote-server,

then this action will fail. The **copy support** command will not create the remote directory.

For example, if the SLX host details are as under:

```
Hostname: SLX-MY-HOST  
Serial-Number: TH000001Q-00001  
Time Stamp: 23-OCT-2021 02:25:36 PM
```

and the remote server details are as under:

```
Hostname: 10.25.37.42  
User-name: scp-user  
Password: password  
Destination Directory: /scp-dir/support-saves/
```

Then the Support Save files are stored in the newly created directory *SLX-MY-HOST_TH000001Q-00001_1023_1425*. If the destination directory */scp-dir/support-save* is not available, then SCP copy will fail.

The path under which you will find the Support Save files will be:

```
/scp-dir/support-saves/SLX-MY-HOST_TH000001Q-00001_1023_1425
```

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.



Warning

If you try to create more than 32 non-root CLI sessions (either SSH or TELNET sessions, or a combination of both sessions), a message will be displayed to close one of the existing sessions to proceed.

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 '^J'.
...
device login:
```

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

Enable or Disable Telnet service

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

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

```
device# configure terminal
```

2. Disable Telnet service on the device.

```
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. Enable Telnet service on the device.

```
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 log in 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 log in.

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

1. Import an SSH client public key to the device.

```
device# certutil import sshkey directory /root/.ssh/ file id_rsa.pub
host 10.20.238.152 login root password pass protocol SCP user admin
```

This example imports the SSH client public key for the admin user from the remote 10.20.238.152 host using the directory and file information for the key and using SCP log-in credentials.

You can also copy the public key directly. For example:

```
device# certutil sshkey user admin pubkey "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQ
Dnim+Ofjx/id3z2jDxXu9DcMuQqVq/
NKi2Lms+q7dA5Dqww8jlrOGawG8tMySOvnB1ZEjVJt1kqNneRi4l6Ot4/7hfd
99rIOPGBF/
NJs6xTLUrQhDgxB78ddTg+6euBtkYLTaATC7kbXGXcO8VVB9+4xrH+0bkvjU9RRvGJguUfdiFKEfIGVOy
t0atdHildmgQ9BE0cO65nc/i9MjMJedBe174/
QT4TxeGeEgaQ57c2AL5It2V4CzrZBDtnixdnHU05w2vmBR61LZIDVT1
fuX/xYxDAm9H8SDpDX8pZlFpQBy/wrkIYPZ/p4OLrUApB/XAJGuJrlNlZLEu9U9MPVM/
root@ldap.hc-fusion.in"
```

After the public key is imported or copied for a user, password-based authentication becomes a fallback option for that particular user. This user can log in using the public key. If a user tries to log in from a device on which the public key is not

present, then the user is prompted for a password. When the public key is removed for the user, only password-based authentication is enabled for that particular user.



Note

When 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 is displayed.

```
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.

This action 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

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

Enable or Disable SSH Service

The SSH service is enabled by default. Disabling 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.

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

```
device# configure terminal
```

2. Disable 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. Enable SSH service on the device.

```
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 can 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 banners on SLX-OS

The SLX-OS device can be configured to display three (3) different greeting messages on user terminals as banners.

- **banner incoming** - Banner displays when the user accesses the SLX-OS device using Telnet or SSH
- **banner motd** - Banner displays when the user attempts to log into the SLX-OS device by entering a user name. This message is displayed before a Telnet / SSH session is established.
- **banner login** - Banner displays when the user successfully logs into the SLX-OS device.

Complete the following steps to set and display a banner.

1. Access global configuration mode.

```
SLX # configure terminal
Entering configuration mode terminal
```

2. Configure the login banner.

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

This example configures a text message on a single line by enclosing the text in double quotation marks (" "). Ensure that you terminate a single line banner with **\n**.

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.



Note

When your banner message is short and is displayed within one line, terminate the message with the **\n** new line character within double quotes (" ").

3. Verify the configured banner.

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

The configured banner is displayed.

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

4. Configure the login banner.

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

This example configures a text message on a single line by enclosing the text in double quotation marks (" ") and ending the message with the "\n" new line character.

The banner can be up to 2048 characters long. To create a multi-line banner, enter the **banner** command followed by the first line of the message. Use the **Esc-m** keys to move to the next line to continue the message. Enter **Ctrl-D** to terminate the input.

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

5. 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\n"
```

The following example is the configuration of the previous steps.

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

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.

Configuring a static ethernet address

You can configure IPv4 and IPv6 static Ethernet network interface addresses in environments where the DHCP service is not available.

Before you configure a static address, connect to the device through the serial console. To configure static Ethernet network interface addresses, perform the following steps:

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

```
device# configure terminal
```

2. Access the interface management mode for the management interface.

```
device(config)# interface Management 1
```

This interface uses the default mgmt-vrf VRF.

3. Disable DHCP.

```
device(config-Management-0)# no ip address dhcp
```

4. Configure the IP address for the management interface.

```
device(config-Management-0)# ip address 10.24.85.81/20
```

5. If you are going to use an IPv6 address, configure the address.

```
device(config-Management-0)# ipv6 address 2001:DB8::69bc:832:e61f:13ff:fe67:4b94/32
```

6. Verify the configuration.

```
device(config-Management-0)# do show running-config interface Management 0
interface Management 0
  no ip address dhcp
  ip address 10.24.85.81/20
  ipv6 address 2001:DB8::69bc:832:e61f:13ff:fe67:4b94/32
!
```

The following example is the configuration of the previous steps.

```
device# configure terminal
device(config)# interface Management 0
device(config-Management-0)# no ip address dhcp
device(config-Management-0)# ip address 10.24.85.81/20
device(config-Management-0)# ipv6 address 2001:DB8::69bc:832:e61f:13ff:fe67:4b94/32
```

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" ]
```

Redundant Management Interface

Introduction

Redundant Management Interface provides fault tolerant management access to remote SLX boxes by providing multiple management access paths to the device. RMI works by pairing the Physical Management Port of the SLX box with one of the physical front panel User Ports. The paired User Port starts in the 'Active-Standby' mode. By default, the Physical Management port is the 'Active' and 'Primary' port and the paired User Port, the 'Standby'. Only one User Port can be configured as the Redundant Management Interface.

Redundant Management Interface uses Linux's native bonding feature to enable the inbuilt Management Port of the SLX box to work with the user configured physical port to provide redundancy.

**Note**

In-band ports bandwidth shall be rate limited as per platform and are unrelated to front-panel port capability.

**Note**

Redundant Management is supported on the SLX 9250/Extreme 8720 and the SLX 9740/Extreme 8820 devices.

eth0 is a logical virtual bridge interface serving both SLX-Linux and TPVM management interfaces. It has a secondary interface **bond0**, which is a Linux LAG in Active-Standby mode, providing fault-tolerance on the Management Path. **bond0** has a preconfigured Primary-Active member, **eth3**. Primary which implies that this member is used and active whenever available. The **eth3** interface represents the physical management interface RJ45 port available on the front panel. For fault tolerance, you can select a standby member from any of the front panel ports, with any native speed. The standby member is used only if the active member becomes faulty.

A user port may be used with a Mellanox Adaptor at 1G Cu SFP or 10G Cu SFP.

To configure a physical front panel port on the device (for example, eth0/15) as a redundant management port, execute the `redundant-management enable` command from within its context. Both the configured redundant management interface ports (eth 0/15 and primary port eth3) must be physically connected to two (2) management LAN switches where these switches share the same default gateway.

Configuration Considerations

- Front panel user port member can have limited bandwidth, irrespective of port's native speed.
- Only one user port can be added as the redundant management interface.
- Across reboot / power cycle, redundancy is enabled only when **redundant-management enable** command is executed when the configuration is replayed back from the last persisted startup configuration.

Configure Redundant Management

The Redundant Management Interface feature provides fault tolerance for the management path.

To configure Redundant Management, use the following commands.



Note

Redundant Management is supported on the SLX 9250/Extreme 8720 and the SLX 9740/Extreme 8820 devices.

1. Ensure interface management is enabled.

```
device# configure
device(config)# interface management 0
device(config-Management-0)# ip address dhcp
device(config-Management-0)# no shut
device(config-Management-0)# exit
```

2. Enable redundant management on an interface/port. RMI is always enabled on one of the physical user ports. To enable RMI on an interface/port:

```
device(config)# interface ethernet 0/15
device(conf-if-eth-0/15)# redundant-management enable
device(conf-if-eth-0/15)# no shut
device(conf-if-eth-0/15)# exit
```

3. Use **show interface management** and **show interface ethernet** commands to verify the configuration.

```
(device)# show interface Management 0
interface Management 0
line-speed actual "1000baseT, Duplex: Full"
oper-status up
ip address "static 10.24.12.89/22"
ip gateway-address 10.24.12.1
ipv6 ipv6-address [ ]
ipv6 ipv6-gateways [ ]
redundant management port 0/15

(device)# show interface ethernet 0/15
Ethernet 0/15 is admin down, line protocol is down (admin down)
Redundant management mode is enabled
Hardware is Ethernet, address is 609c.9f5a.a35f
  Current address is 609c.9f5a.a35f
Pluggable media not present
Description: Insight port
Interface index (ifindex) is 202350592 (0xc0fa000)
MTU 9216 bytes
Maximum Speed      : 10G
LineSpeed Actual    : Nil
LineSpeed Configured : Auto, Duplex: Full
Priority Tag disable
Forward LACP PDU: Disable
Route Only: Disabled
Tag-type: 0x8100
Last clearing of show interface counters: 00:01:13
Queueing strategy: fifo
FEC Mode - Disabled
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
Route-Only Packets Dropped: 0
Time since last interface status change: 00:01:13

```

Configure Redundant Management with a Mellanox Adapter

The Redundant Management Interface feature provides fault tolerance for the management path.

The following procedure configures redundant management on port 0/15 with a Mellanox Adapter.

1. Configure port 0/15.

```

device# configure
device(config)# hardware
device(config-hardware)# connector 0/15
device(config-connector-0/15)#
device#

```

2. Configure breakout mode of 4x1G. (only on SLX 9250/Extreme 8720 devices)

```

device(config-connector-0/15)# breakout mode 4x1G
device(config-connector-0/15)# end
device#

```

3. Configure breakout mode of 4x10G. (on both SLX 9250/Extreme 8720 and SLX 9740/Extreme 8820 devices)

```

device(config-connector-0/15)# breakout mode 4x10G
device(config-connector-0/15)# end
device#

```

4. Configure redundant management on port 0/15.

```

device# configure
device(config)# interface Ethernet 0/15:1
device(config-hardware)# connector 0/15
device(conf-if-eth-0/15:1)# redundant-management enable
device(conf-if-eth-0/15:1)# no shut
device(conf-if-eth-0/15:1)# end
device#

```

When using the Mellanox adaptor, only the first member of the breakout board can be used as the Redundant Management Interface. This is shown in the above example.

The following example displays the Redundant Management Interface configuration on SLX 9250/Extreme 8720 devices with 4x10G.

```

SLX# show running-config hardware connector 0/15 breakout mode
hardware
connector 0/15
  breakout mode 4x10g
!

```



```

SLX# show running-config interface Ethernet 0/15:1
interface Ethernet 0/15:1
redundant-management enable
no shutdown
!

SLX# show interface ethernet 0/15:1
Ethernet 0/15:1 is up, line protocol is up (connected)
Redundant management mode is enabled
Hardware is Ethernet, address is f46e.95a1.f826
  Current address is f46e.95a1.f826
Fixed Copper RJ45 Media Present
Interface index (ifindex) is 201589248 (0xc040200)
MTU 9216 bytes
IP MTU 1500 bytes
Maximum Speed      : 10G
LineSpeed Actual   : 10000 Mbit
LineSpeed Configured : Auto, Duplex: Full
Priority Tag disable
Forward LACP PDU: Disable
Route Only: Disabled
Tag-type: 0x8100
Last clearing of show interface counters: 02:45:29
Queueing strategy: fifo
FEC Mode - Disabled
Receive Statistics:
  81295 packets, 6130007 bytes
  Unicasts: 1212, Multicasts: 71717, Broadcasts: 8363
  64-byte pkts: 62703, Over 64-byte pkts: 17088, Over 127-byte pkts: 545
  Over 255-byte pkts: 606, Over 511-byte pkts: 117, Over 1023-byte pkts: 233
  Over 1518-byte pkts(Jumbo): 0
  Runt: 0, Jabbers: 0, CRC: 0, Overruns: 0
  Errors: 0, Discards: 0
Transmit Statistics:
  742 packets, 66702 bytes
  Unicasts: 426, Multicasts: 316, Broadcasts: 0
  Underruns: 0
  Errors: 0, Discards: 0
Rate info:
  Input 0.005267 Mbits/sec, 9 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: 00:00:44

```

Force 1G connection speed on physical interfaces

In some scenarios, it is desirable that connections on a specific interface be restricted to a lower speed than the maximum supported speed. Such scenarios occur when the device that is connected to the interface is unable to connect at the highest speed possible for that interface.

This speed setting is only supported on the 1G/10G copper ports (ports 1-48) of the SLX 9150-48XT and Extreme 8520-48XT devices to enable the user to force 1000Mbps (1G) speed setting on these interfaces in both full-duplex and half-duplex mode. The master/slave setting allows the interface to act as either a clock master or a slave in the forced speed setting.

The following speed configurations are available:

- 1000-auto-full-duplex
- 1000-master-full-duplex
- 1000-master-half-duplex
- 1000-slave-full-duplex
- 1000-slave-half-duplex

The **1000-auto-full-duplex** command enables the configuration of the ports to be set at 1G speed and in full duplex mode.

To reset the above setting to the default setting of 1G/10G AN mode, use either the **speed auto** or the **no speed** command.

When using the master/slave configuration, ensure that the peer port is configured with the complimentary configuration for the port to come up. For example:

- If the local port is configured as **1000-master-full-duplex**, then the peer port must be configured as **1000-slave-full-duplex**.
- If the local port is configured as **1000-slave-half-duplex**, then the peer port must be configured as **1000-master-half-duplex**.
- If the local port is configured as **1000-auto-full-duplex**, then the peer port must be configured for auto-negotiation for the link to be established.

The **show interface ethernet** command is also enhanced to show the configured speed for the interface.

```
SLX(conf-if-eth-0/1-2)# do show interface ethernet 0/1
Ethernet 0/1 is up, line protocol is up (connected)
Hardware is Ethernet, address is f064.26f2.d80c
Current address is f064.26f2.d80c
Fixed Copper RJ45 Media Present
Interface index (ifindex) is 201334784 (0xc002000)
MTU 9216 bytes
Maximum Speed : 10G
LineSpeed Actual : 1000 Mbit
LineSpeed Configured : 1G forced master, Duplex: Half
Priority Tag disable
Forward LACP PDU: Disable
Route Only: Disabled
```

The following example displays the forced 1G configuration on the Extreme 8520-48XT device

```
8520-48XT# show interface ethernet 0/36
Ethernet 0/36 is up, line protocol is up (connected)
Hardware is Ethernet, address is f064.26f4.4c2f
Current address is f064.26f4.4c2f
Fixed Copper RJ45 Media Present
Interface index (ifindex) is 201621504 (0xc048000)
MTU 9216 bytes
Maximum Speed : 10G
LineSpeed Actual : 1000 Mbit
LineSpeed Configured : 1G auto full, Duplex: Full
Priority Tag disable
```

```
Forward LACP PDU: Disable
Route Only: Disabled
```

The following example shows how to configure the Force 1G speed configuration on an Extreme 8520-48XT device.

```
8520-48XT(config)# interface Ethernet 0/36
8520-48XT(conf-if-eth-0/36)# speed ?
Possible completions:
 100                100Mbps
1000               1Gbps
1000-auto          1Gbps AN (802.3 Clause 37 Auto-Negotiation)
1000-auto-full-duplex 1Gbps Auto Negotiation for copper cable
1000-master-full-duplex 1Gbps - Set as Master with Full Duplex
1000-master-half-duplex 1Gbps - Set as Master with Half Duplex
1000-slave-full-duplex 1Gbps - Set as Slave with Full Duplex
1000-slave-half-duplex 1Gbps - Set as Slave with Half Duplex
10000              10Gbps
25000              25Gbps
40000              40Gbps
100000             100Gbps
auto               Auto Detection (default)

8520-48XT(conf-if-eth-0/36)# speed 1000-auto-full-duplex
8520-48XT(conf-if-eth-0/36)# no shutdown
8520-48XT(conf-if-eth-0/36)# end
8520-48XT# show running-config interface Ethernet 0/36
interface Ethernet 0/36
speed 1000-auto-full-duplex
no shutdown
!
8520-48XT#
```

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.

```
SLX# show system
Stack MAC                : 60:9c:9f:de:29:14

-- UNIT 0 --
Unit Name                 : SLX
Up Time                   : up 11 days
Current Time              : 10:41:45 GMT
SLX-OS Version            : 20.4.1
```

```
Jumbo Capable      : yes
Burned In MAC      : 60:9c:9f:46:e2:06
Management IP      : 10.20.131.53
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/Extreme 8720 port management includes the following:
 - Supports 32 ports of 40G and 100G.
 - Ports may be broken out into 4x10G or 4x25G.
- SLX 9150-48Y/Extreme 8520-48Y 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).
- SLX 9150-48XT/Extreme 8520-48XT 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 in 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
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 0/4
```

3. Configure port flap dampening.

```
device(config-if-eth-0/4)# 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 0/4
device(config-if-eth-0/4)# 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 0/2
```

3. Configure the port transition hold timer.

```
device(conf-if-eth-0/2)# 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-0/2)# do show running-config interface ethernet 0/2
interface Ethernet 0/2
...
delay-link-event 2 down
no shutdown
!
```

The following example shows the steps in the previous configuration.

```
device# configure terminal
device(config)# interface Ethernet 0/2
device(conf-if-eth-0/2)# 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(config-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(config-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(config-if-eth-0/1)# no shutdown
```

7. Verify the configuration for the interface.

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

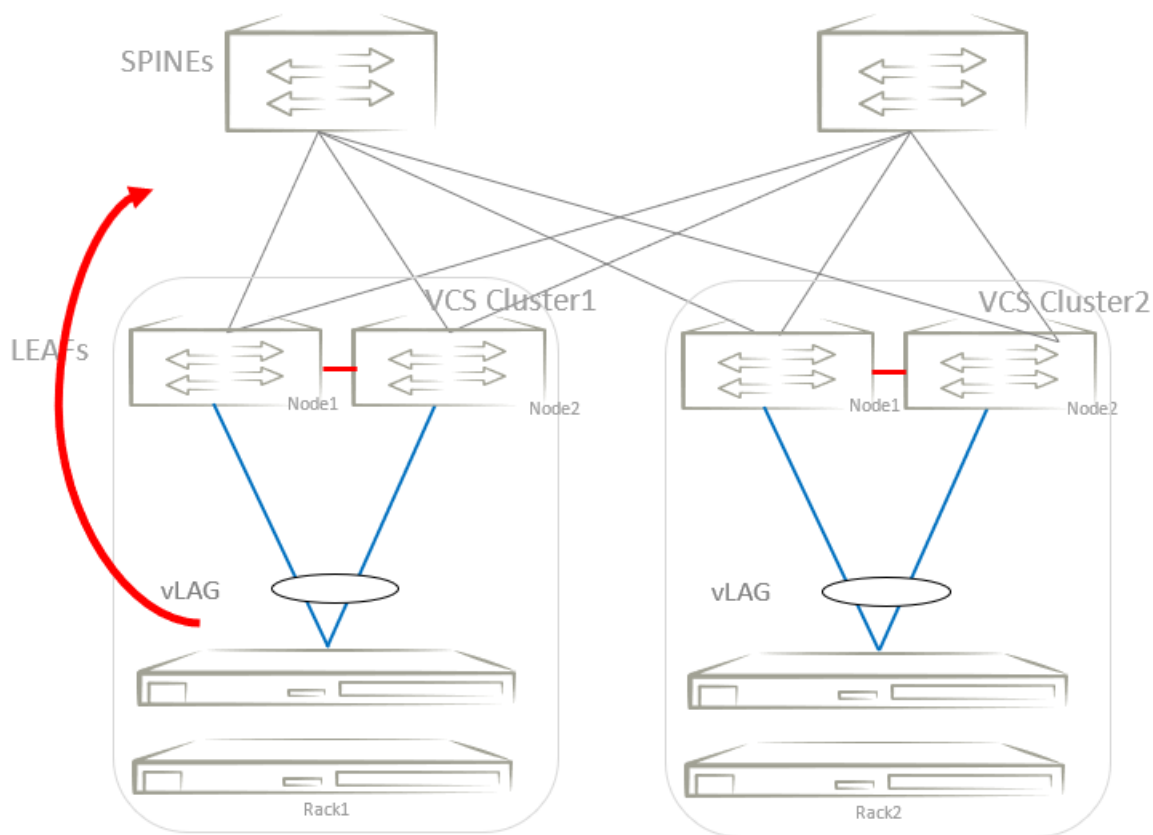


Figure 1: Scenario 1

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.

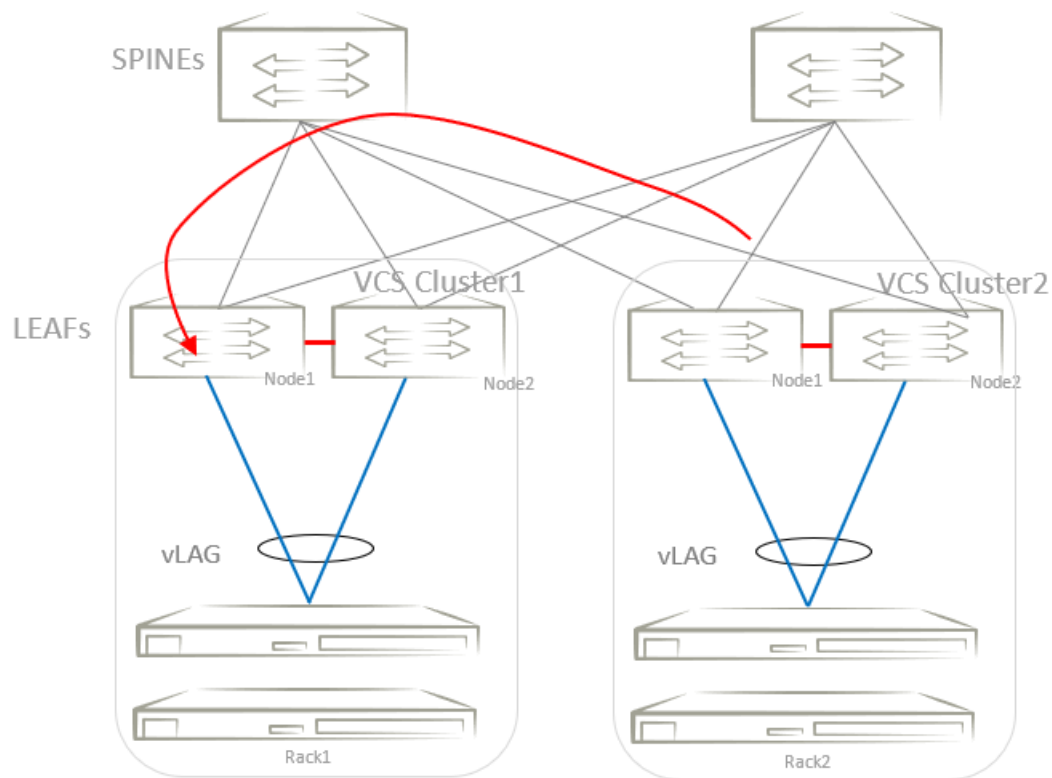


Figure 2: Scenario 2

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 SLX<model> where *model* is the model name of 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

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

Service	VRF-related command	Additional information
sFlow	<pre>[no] sflow collector ipv4/ipv6 address port-number [use-vrf vrf-name] [no] sflow source-interface interface-type interface-number</pre>	<p>Uses sFlow UDP IPv4 or IPv6. In the case of the sflow source-interface command, the VRF of the specified interface is used.</p> <p>Note: Any given interface can belong to only one VRF at any given time.</p>
SSH Client	<pre>ssh { IP_address hostname } [-l] remote user/login name [vrf vrf-name]</pre>	VRF is optional; default-vrf is used by default.
SNMP notification	<pre>snmp-server host ip-address [use-vrf vrf-name] snmp-server v3host ip-address [use-vrf vrf-name]</pre>	Uses SNMP UDP IPv4 or IPv6. By default, mgmt-vrf is used to send the SNMP notifications.
Support save	<pre>copy support { ftp scp } [use-vrf vrf-name] ...</pre>	Uses TCP IPv4 or IPv6. By default, mgmt-vrf is used and a user-defined VRF is optional.
TACACS+	<pre>tacacs-server host host-name [use-vrf vrf-name]</pre>	Uses TCP IPv4 or IPv6
Telnet client	<pre>telnet IP_address hostname [vrf vrf-name]</pre>	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.

Heartbeat between SLX and EFA

This topic discusses the need for a heartbeat between SLX and EFA and how the state of the SLX can be modified based on the heart beats received from EFA.

This feature enables changing the state of the SLX OS device depending on the availability of heartbeats from an external management application, specifically, Extreme Networks's *Extreme Fabric Automation* (EFA) application.

Once this feature is enabled on the SLX device, it starts to listen to periodic heartbeats from the remote EFA application. When the SLX device - that is listening to a heartbeat - does not receive these heartbeat messages for a configured time interval, it can declare itself as being *Out Of Service* and perform a set of pre-defined actions.

When this feature is not enabled, the SLX device ignores any heartbeats it receives. It is considered to be in the *Management Operational State - Down* state.

When enabled for the first time

When the feature is enabled for the first time, the SLX device starts with a state of *Management Operational State - Down*. It enables itself and starts to listen for the heartbeats from EFA. When it receives the first heartbeat, it brings itself up to *Management Operational State - Up*.

When heartbeat messages are not received

When the device is in the *Management Operational State - Up* state, it keeps on listening for heartbeat messages. When any heartbeat message is not received for a pre-defined time interval, it enters itself into *Management Operational State - Down*. It then can perform a few pre-defined actions. After completing these actions, the device places itself in *Maintenance Mode*.

After a reboot

When the SLX device comes up after a reboot, it enters itself into the state it was in before the reboot. If the Management Heartbeat feature was enabled, the device starts listening for the first heartbeat message from EFA and on receiving it enters into *Management Operational State - Up* state.

Transitioning between Management Operational State - Down and Up states.

If the SLX device enters the *Management Operational State - Down* state after not having receiving heartbeat messages from EFA, it remains in the *Admin Up State*. While in this state, it continues to listen for the first heartbeat message from EFA and on receiving it, enters into the *Management Operational State - Up* state.

When the SLX device is *Admin Up State* due to it being in *Management Operational State - Down*, an administrator May change its state to *Admin Down State*.

Heartbeat Setting

Describes the settings for enabling heartbeat message monitoring by SLX devices.

Heartbeat monitoring is configured from a sub-context under *Global Configuration*.

1. From within the *Global Configuration* context, issue the `management-heartbeat manager` command.

The context changes to `SLX(config-management-heartbeat-manage) #` prompt.

```
SLX(config) # management-heartbeat manager
SLX(config-management-heartbeat-manage) #
```

2. Set the heartbeat message listening threshold value. Use the `threshold-value <threshold-value-in-minutes>` command to configure this value.

Use the `[no] threshold-value` command to reset the threshold timer value.

This is the time duration for which the SLX device listens for a heartbeat message from EFA and on expiry of this threshold time, the device goes into a *Management Operational State - Down* state.

This example sets the threshold value to thirty (30) minutes. The default value for this command is five (5) minutes.

```
SLX(config-management-heartbeat-manage) # threshold-value 30
```

When the `no threshold-value` command is used, the threshold is reverted to the default value of 5 minutes.

```
SLX(config-management-heartbeat-manage) # no threshold-value
```

3. Set the actions that need to be performed when the SLX device does not receive any heartbeat message for the threshold duration. Use the `action` command to do so.

The `action` command can set the SLX device into management mode only. If no action is needed, the use the `no-action` command.

When the SLX device does not receive any heartbeat message from the EFA, then it can either enter into *Management Operational State - Down* or perform no action.

The following example is for configuring an action of setting the SLX device to *Management Operational State - Down*.

```
SLX(config-management-heartbeat-manage) # action maintenance-mode-enable
```

The following example is for taking no action.

```
SLX(config-management-heartbeat-manage) # action no-action
```

4. Verify the configuration using the `show management-heartbeat-manager` command.

The command displays the current heartbeat manager configuration.

```
SLX(config-management-heartbeat-manage) # show management-heartbeat-manager
Admin state: down
Operational state: up
Threshold time: 30 minutes
Action: Maintenance mode enable
Time to last heartbeat: 4 minutes

SLX(config-management-heartbeat-manage) #
```

5. Use the `enable` command to enable Heartbeat monitoring on the SLX device.

```
SLX(config-management-heartbeat-manage) # enable
SLX(config-management-heartbeat-manage) # show management-heartbeat-manager
Admin state: up
Operational state: up
Threshold time: 30 minutes
Action: Maintenance mode enable
Time to last heartbeat: 30 minutes
```

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:

- SLX 9540
- SLX 9640
- SLX 9250
- SLX 9150-48Y
- SLX 9150-48XT
- SLX 9740-40C
- SLX 9740-80C
- Extreme 8720
- Extreme 8520-48Y
- Extreme 8520-48XT
- Extreme 8820-40C
- Extreme 8820-80C

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.
- For secure ZTP, DHCP server must be configured with option 43 to set HTTPS configuration.
- ZTP is triggered on a new device by means of *Open Network Install Environment* (ONIE), by 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, HTTP, and HTTPS to fetch ZTP configurations, scripts, switch startup configurations, and firmware.

The DHCP and FTP/HTTP/HTTPS 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/HTTPS 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.

ZTP over HTTPS

ZTP supports HTTPS from SLX-OS version 20.7.2 onwards. The following additional conditions apply when using ZTP over HTTPS.

- If Option 43 is not configured on the DHCP server, the SLX-OS device will skip using HTTPS. It will use FTP/HTTP to download SLX-OS image, configuration files, and scripts.
- On the DHCP server, the path to the CA certificate of the HTTPS server is encoded within Option 43 in **HEX** format. This **HEX** value is then sent through DHCP message.
- The client, on receipt of this DHCP message, tries to verify if the path to the HTTPS server's certificate is valid. If the passed URL is invalid or the certificate has expired, the ZTP process will go into an infinite loop and a manual intervention may be required.
- If Option 43 was sent by the DHCP Server, the client then downloads the Server's CA Certificate and uses it to create a secured connection to download the various scripts, config files, and SLX image.

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 or 43 configured over a management interface and all in-band port interfaces.

Option 43 is used when the DHCP server supports HTTPS protocol.
 - d. Selects one interface not used before to assign the DHCP IP address.
 - e. Downloads the ZTP configuration file from the FTP/HTTP/HTTPS server depending on the configured option.

- 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.

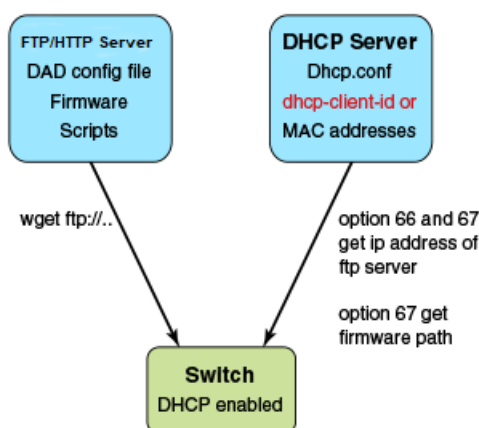


Figure 3: ZTP configuration

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.

Option 43 is used to share the URL for the HTTPS server's CA certificate so that clients can use HTTPS to securely connect to. This URL is **HEX** encoded.

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/HTTPS server

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

ZTP configuration script

The ZTP process can run the script to set up the device configuration automatically. At present, 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 settings in 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
    "dtBgNTAoqZmwV5c4SueybjOvhe60Iggac1uQrzGBv504X4nIEBEEGWRf0lCnbFhuIJXGExNBjDdNSqgBMeNI8w=="
    ;
};

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;
    option tftp-server-name "192.168.0.1";
    option bootfile-name "/config/ztp.cfg";
    option vendor-encapsulated-options
    68:74:74:70:73:3a:2f:2f:35:2e:35:2e:35:2e:31:3a:34:34:33:2f:73:65:72:76:65:72:2e:63:72:74;
    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.

Table 7: ZTP configuration file definitions (continued)

Variable description	Description
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.
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 for FTP/HTTP.

```
device# dhcp ztp log
```

```

ZTP, Thu Apr 10 12:48:51 2025, ===== ZTP start =====
ZTP, Thu Apr 10 12:48:51 2025, disable raslog
ZTP, Thu Apr 10 12:48:51 2025, CLI is ready
ZTP, Thu Apr 10 12:49:19 2025, inband ports are enabled
ZTP, Thu Apr 10 12:49:19 2025, serial number = 771232X1750017
ZTP, Thu Apr 10 12:49:19 2025, model name = AS7712-32X
ZTP, Thu Apr 10 12:49:19 2025, use both management interface and inband interfaces
ZTP, Thu Apr 10 12:49:19 2025, checking inband interfaces link status
ZTP, Thu Apr 10 12:49:19 2025, find link up on interfaces: eth0
ZTP, Thu Apr 10 12:49:19 2025, start dhcp process on interfaces: eth0
ZTP, Thu Apr 10 12:49:20 2025, interface eth0 receives dhcp response
ZTP, Thu Apr 10 12:49:20 2025, ping server 192.169.0.1
ZTP, Thu Apr 10 12:49:21 2025, ping succeed
ZTP, Thu Apr 10 12:49:21 2025, download ZTP config file from https://192.169.0.1/config/
ztp.conf
ZTP, Thu Apr 10 12:49:21 2025, download ZTP config file from http://192.169.0.1/config/
ztp.conf
ZTP, Thu Apr 10 12:49:21 2025, receive ZTP configuration file [ztp.conf]
ZTP, Thu Apr 10 12:49:21 2025, interface eth0 connectivity test pass
ZTP, Thu Apr 10 12:49:21 2025, download switch config file [startup.cfg]
ZTP, Thu Apr 10 12:49:21 2025, ZTP configuration sanity check pass
ZTP, Thu Apr 10 12:49:22 2025, skip firmware upgrade
ZTP, Thu Apr 10 12:49:38 2025, replay config file...
ZTP, Thu Apr 10 12:50:25 2025, commit configuration
ZTP, Thu Apr 10 12:50:25 2025, ZTP succeed
ZTP, Thu Apr 10 12:50:25 2025, enable raslog
ZTP, Thu Apr 10 12:50:25 2025, ===== 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 displays current ZTP progress for HTTPS.

```

device# dhcp ztp log
ZTP, Thu Apr 10 13:53:15 2025, ===== ZTP start =====
ZTP, Thu Apr 10 13:53:15 2025, disable raslog
ZTP, Thu Apr 10 13:53:15 2025, CLI is ready
ZTP, Thu Apr 10 13:53:31 2025, inband ports are enabled
ZTP, Thu Apr 10 13:53:31 2025, serial number = 1927Q-20908
ZTP, Thu Apr 10 13:53:31 2025, model name = SLX9150-48XT
ZTP, Thu Apr 10 13:53:31 2025, use both management interface and inband interfaces
ZTP, Thu Apr 10 13:53:32 2025, checking inband interfaces link status
ZTP, Thu Apr 10 13:53:32 2025, find link up on interfaces: eth0
ZTP, Thu Apr 10 13:53:32 2025, start dhcp process on interfaces: eth0
ZTP, Thu Apr 10 13:53:42 2025, retry in 10 seconds
ZTP, Thu Apr 10 13:53:52 2025, inband ports are enabled
ZTP, Thu Apr 10 13:53:52 2025, serial number = 1927Q-20908
ZTP, Thu Apr 10 13:53:52 2025, model name = SLX9150-48XT
ZTP, Thu Apr 10 13:53:52 2025, use both management interface and inband interfaces
ZTP, Thu Apr 10 13:53:52 2025, checking inband interfaces link status
ZTP, Thu Apr 10 13:54:44 2025, find link up on interfaces: eth0 Eth0.23
ZTP, Thu Apr 10 13:54:44 2025, start dhcp process on interfaces: eth0 Eth0.23
ZTP, Thu Apr 10 13:54:46 2025, interface Eth0.23 receives dhcp response
ZTP, Thu Apr 10 13:54:48 2025, config ip address 5.5.5.12/24 on interface Eth0.23
ZTP, Thu Apr 10 13:54:55 2025, ping server 5.5.5.1
ZTP, Thu Apr 10 13:54:56 2025, ping succeed
Downloaded through wget command https://5.5.5.1:443/server.crt
Downloaded through wget command yes server.crt.
Downloaded server.crt to /etc/ssl/certs.
ZTP, Thu Apr 10 13:54:56 2025, download ZTP config file from https://5.5.5.1/config/
ztp.cfg
ZTP, Thu Apr 10 13:54:56 2025, receive ZTP configuration file [ztp.cfg]

```

```

ZTP, Thu Apr 10 13:54:56 2025, interface Eth0.23 connectivity test pass
ZTP, Thu Apr 10 13:54:59 2025, firmware upgrade sanity check passed
ZTP, Thu Apr 10 13:54:59 2025, ZTP configuration sanity check pass
ZTP, Thu Apr 10 13:54:59 2025, start firmware upgrade...
ZTP, Thu Apr 10 14:03:51 2025, ===== ZTP continue =====
ZTP, Thu Apr 10 14:03:51 2025, disable raslog
ZTP, Thu Apr 10 14:03:51 2025, CLI is ready
ZTP, Thu Apr 10 14:03:52 2025, start firmware commit
ZTP, Thu Apr 10 14:05:22 2025, firmware upgrade succeed.
ZTP, Thu Apr 10 14:05:47 2025, commit configuration
ZTP, Thu Apr 10 14:05:47 2025, ZTP succeed
ZTP, Thu Apr 10 14:05:47 2025, enable raslog
ZTP, Thu Apr 10 14:05:47 2025, ===== ZTP completed =====

device#

```

The following action 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/HTTPS server. Switch 1 acts as a router for Switch 2 to reach the DHCP or FTP/HTTP/HTTPS 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.

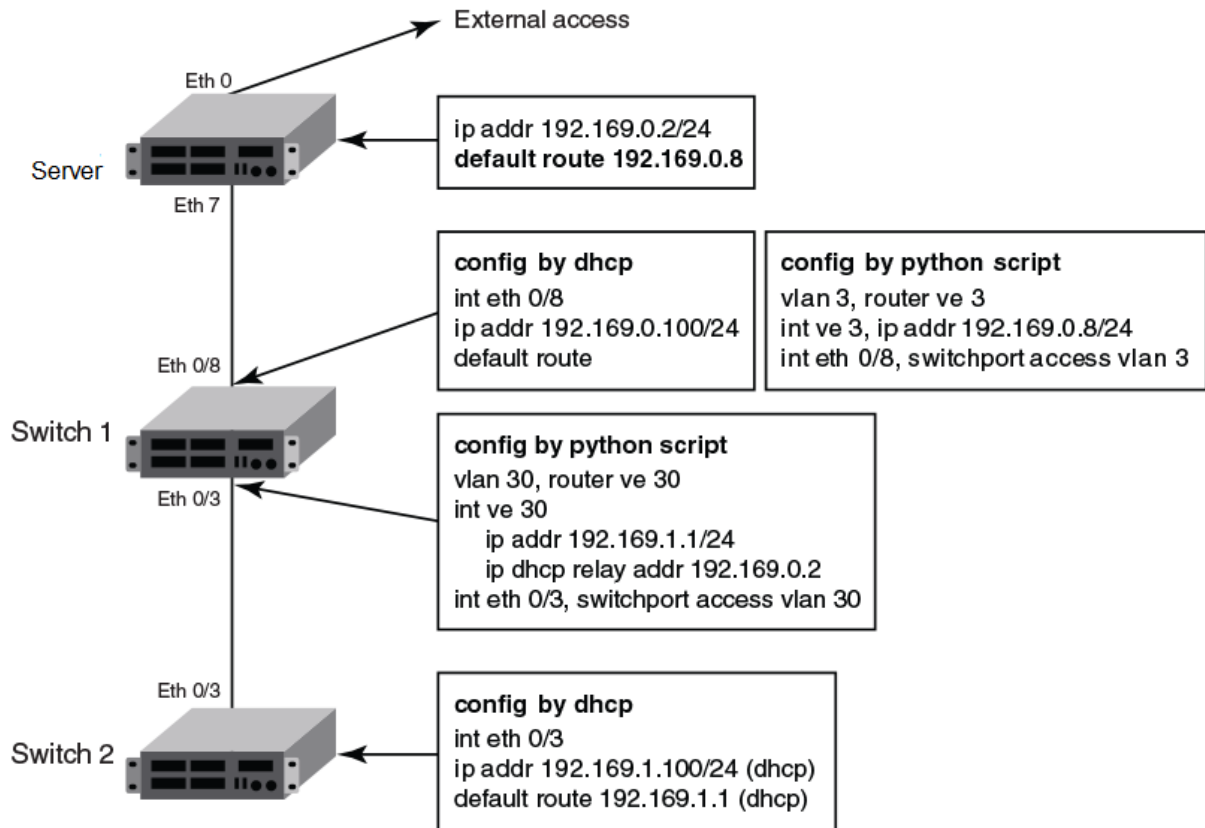


Figure 4: ZTP two-node topology



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;
    option vendor-encapsulated-options
68:74:74:70:73:3a:2f:2f:35:2e:35:2e:35:2e:31:3a:34:34:33:2f:73:65:72:76:65:72:2e:63:72:74;
}
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/freedoml_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 for FTP/HTTP 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]

```

The following displays current ZTP progress for HTTPS.

```

device# dhcp ztp log
ZTP, Thu Apr 10 13:53:15 2025, ===== ZTP start =====
ZTP, Thu Apr 10 13:53:15 2025, disable raslog
ZTP, Thu Apr 10 13:53:15 2025, CLI is ready
ZTP, Thu Apr 10 13:53:31 2025, inband ports are enabled
ZTP, Thu Apr 10 13:53:31 2025, serial number = 1927Q-20908
ZTP, Thu Apr 10 13:53:31 2025, model name = SLX9150-48XT
ZTP, Thu Apr 10 13:53:31 2025, use both management interface and inband interfaces
ZTP, Thu Apr 10 13:53:32 2025, checking inband interfaces link status
ZTP, Thu Apr 10 13:53:32 2025, find link up on interfaces: eth0
ZTP, Thu Apr 10 13:53:32 2025, start dhcp process on interfaces: eth0
ZTP, Thu Apr 10 13:53:42 2025, retry in 10 seconds
ZTP, Thu Apr 10 13:53:52 2025, inband ports are enabled
ZTP, Thu Apr 10 13:53:52 2025, serial number = 1927Q-20908
ZTP, Thu Apr 10 13:53:52 2025, model name = SLX9150-48XT
ZTP, Thu Apr 10 13:53:52 2025, use both management interface and inband interfaces
ZTP, Thu Apr 10 13:53:52 2025, checking inband interfaces link status
ZTP, Thu Apr 10 13:54:44 2025, find link up on interfaces: eth0 Eth0.23
ZTP, Thu Apr 10 13:54:44 2025, start dhcp process on interfaces: eth0 Eth0.23
ZTP, Thu Apr 10 13:54:46 2025, interface Eth0.23 receives dhcp response
ZTP, Thu Apr 10 13:54:48 2025, config ip address 5.5.5.12/24 on interface Eth0.23
ZTP, Thu Apr 10 13:54:55 2025, ping server 5.5.5.1
ZTP, Thu Apr 10 13:54:56 2025, ping succeed
Downloaded through wget command https://5.5.5.1:443/server.crt
Downloaded through wget command yes server.crt.

```

```

Downloaded server.crt to /etc/ssl/certs.
ZTP, Thu Apr 10 13:54:56 2025, download ZTP config file from https://5.5.5.1/config/
ztp.cfg
ZTP, Thu Apr 10 13:54:56 2025, receive ZTP configuration file [ztp.cfg]
ZTP, Thu Apr 10 13:54:56 2025, interface Eth0.23 connectivity test pass
ZTP, Thu Apr 10 13:54:59 2025, firmware upgrade sanity check passed
ZTP, Thu Apr 10 13:54:59 2025, ZTP configuration sanity check pass
ZTP, Thu Apr 10 13:54:59 2025, start firmware upgrade...
ZTP, Thu Apr 10 14:03:51 2025, ===== ZTP continue =====
ZTP, Thu Apr 10 14:03:51 2025, disable raslog
ZTP, Thu Apr 10 14:03:51 2025, CLI is ready
ZTP, Thu Apr 10 14:03:52 2025, start firmware commit
ZTP, Thu Apr 10 14:05:22 2025, firmware upgrade succeed.
ZTP, Thu Apr 10 14:05:47 2025, commit configuration
ZTP, Thu Apr 10 14:05:47 2025, ZTP succeed
ZTP, Thu Apr 10 14:05:47 2025, enable raslog
ZTP, Thu Apr 10 14:05:47 2025, ===== ZTP completed =====

device#

```



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.

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 the Extreme 8720, Extreme 8520, SLX 9250, and SLX 9150 devices, TCAM banks are referred to as *slices*.

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.

The following table is applicable for Extreme 8820, SLX 9740, SLX 9640, and SLX 9540 devices:

Table 8: 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 Extreme 8720, Extreme 8520, SLX 9250, and SLX 9150, 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 Extreme 8820, SLX 9740, SLX 9640, and SLX 9540.

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.
- 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 (on SLX 9540, SLX 9640, SLX 9740, and Extreme 8820)

For Extreme 8820, SLX 9740, SLX 9640, and SLX 9540 devices, 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 5: TCAM-entries available per profile (Extreme 8820, SLX 9740, SLX 9640, and SLX 9540 devies)

* 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.

TCAM Profile Scaling (SLX 9150, SLX 9250, Extreme 8520, and Extreme 8720 devices)

For Extreme 8720, Extreme 8520, SLX 9250, and SLX 9150 devices, 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 6: 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.

Table 9: TCAM Profile Feature Support (SLX 9740/Extreme 8820)

TCAM Profile	Feature	Default	IPv6 Optimised
INGRESS	Ingress L2 MAC ACLs	Supported	Supported
	Ingress IPv4 ACLs (ACLs, PBR, RACL, RL, RACL-RL, v4Broadcast ACL)	Supported	Supported
	Ingress IPv6 ACLs (ACLs, PBR, RL, RACL, RACL-RL)	Supported	Supported
	VLL(PWE + VXLAN)	Supported	
	BUM RL, PORT RL, VLAN RL + BD RL	Supported	
	Tunnel	Supported	Supported
	XC stat	Supported	Supported
EGRESS	Egress Ipv6 acl	Not Supported	Supported
	Egress L2 ACL (ACL, RL)	Supported	Supported
	Egress IPv4 ACL	Supported	Supported

For scale information, refer *Scale and Standards Matrix* document for this version.

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 Extreme 8820, SLX 9740, SLX 9640, and SLX 9540 devices.

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 10: TCAM-sharing support matrix (only for 9540 and 9640)

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
border-routing	0	No	No	No	No	No
layer2-ratelimit	0	No	No	No	No	No

Table 10: TCAM-sharing support matrix (only for 9540 and 9640) (continued)

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
multicast-profile	3	Yes	Yes	Yes	No	No
vxlان-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 11: 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
Counter-profile-5	Egress Rate Limit optimized, support up to 32K VOQs counting
Counter-profile-6	Egress VE stats support

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 12: 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 13: 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
```

Rebooting into Maintenance Mode

The **config-system-maintenance** command provides two methods for you to access the maintenance mode. The first is to use the **enable** parameter, which puts the device into maintenance mode after a specified convergence time. The second method, uses the **enable-on-reboot** command, allowing the device to enter maintenance mode upon reboot. Use of this command does not disturb the current operation of the device.

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.

Use the **enable-on-reboot** command to enable the device to come up in maintenance mode after a reboot. This process allows any network errors detected with Extreme Fabric Automation (EFA) to be addressed. After the errors have been resolved, the device can be added back to the network.

The following example enables system reboot into maintenance mode

```
device# configure terminal
device(config)# system
device(config-system)# maintenance
device(config-system-maintenance)# enable-on-reboot
```

Use the **enable** parameter with a specified convergence time, to put the device into maintenance mode without a reboot.

```
device# configure terminal
device(config)# system
device(config-system)# maintenance
device(config-system-maintenance)# enable
device(config-system-maintenance)# convergence-time 120
```

Support for OpenConfig Telemetry

Introduction

OpenConfig is a vendor-neutral, model-driven network management specification, where data models are used for both configuration as well as retrieving operational state of the network across platforms.

OpenConfig proposes to use gNMI (gRPC Network Management Interface) framework as the network management protocol for configuration, data retrieval, and real-time network monitoring support. gNMI is a gRPC based protocol developed by Google™. It provides mechanisms to modify and retrieve configuration information from target devices. It also provides the ability to generate and control telemetry streams from these target devices to a data collection system.

SLX can at present only fetch some operational state for a small set of modules. It does not support setting the configuration of these modules which support gNMI.

SLX supports fetching operational state for the following modules:

1. Platform
2. Interface
3. BGP

For each module, of the large amount of information that can be fetched, we support a small set. The following section lists the operational state information that can be fetched for each module.

OpenConfig-Interface YANG Module

Describes the information that can be fetched from the OpenConfig-Interface Yang Module

For the *Interface Module*, the following operational state information can be fetched:

```
module: operconfig-interfaces
path: /interfaces/interface[name=<ifname>]/state
```

```

module: operconfig-interfaces
  +--rw interfaces
    +--rw interface* [name]
      +--rw name --> ../config/name
      | +--ro name? string
      | +--ro mtu? uint16
      | +--ro loopback-mode? boolean
      | +--ro description? string
      | +--ro enabled? boolean
      | +--ro ifindex? uint32
      | +--ro counters
      | | +--ro in-octets? oc-yang:counter64
      | | +--ro in-unicast-pkts? oc-yang:counter64
      | | +--ro in-broadcast-pkts? oc-yang:counter64
      | | +--ro in-multicast-pkts? oc-yang:counter64
      | | +--ro in-discards? oc-yang:counter64
      | | +--ro in-errors? oc-yang:counter64
      | | +--ro in-fcs-errors? oc-yang:counter64
      | | +--ro out-octets? oc-yang:counter64
      | | +--ro out-unicast-pkts? oc-yang:counter64
      | | +--ro out-broadcast-pkts? oc-yang:counter64
      | | +--ro out-multicast-pkts? oc-yang:counter64
      | | +--ro out-discards? oc-yang:counter64
      | | +--ro out-errors? oc-yang:counter64

```

OpenConfig-BGP Yang Module

For the *BGP Module*, the following operational state information can be fetched:

```

module: openconfig-bgp
path: /bgp/neighbors/neighbor[neighbor-address=<nAddress>]

module: openconfig-bgp
  +--rw bgp
    +--rw neighbors
      | +--rw neighbor* [neighbor-address]
      | | +--rw neighbor-address -> ../config/neighbor-address
      | | | +--ro neighbor-address? oc-inet:ip-address
      | | +--rw transport
      | | | +--ro state
      | | | | +--ro local-port? oc-inet:port-number
      | | | | +--ro remote-port? oc-inet:port-number
      | | +--rw use-multiple-paths
      | | | +--ro state
      | | | | +--ro enabled? boolean
      | | +--rw ebgp
      | | | +--ro state
      | | | | +--ro allow-multiple-as? boolean

```

OpenConfig-Platform Yang Module

For the *Platform Module*, the following operational state information can be fetched:

```

module: openconfig-platform
path: /components/component[name=<cmpntName>]/state

```

```

+--rw components
  +--rw component* [name]
    +--rw name                               -> ../config/name
    +--ro state
      | +--ro name?                          string
      | +--ro id?                           string
      | +--ro mfg-name?                     string
      | +--ro hardware-version?             string
      | +--ro firmware-version?             string
      | +--ro software-version?             string
      | +--ro serial-no?                    string
      | +--ro part-no?                      string
      | +--ro removable?                    boolean
      | +--ro empty?                        boolean
      | +--ro parent?                       -> ../../config/name
      | +--ro temperature
        | | +--ro alarm-status?              boolean
        | | +--ro alarm-threshold?          uint32

```

Enabling OpenConfig Telemetry Support

This topic describes the steps to enable OpenConfig Telemetry Support on SLX-OS.

It is assumed that your infrastructure is set up with a gNMI client. Its configuration is beyond the scope of this document.

SLX-OS provides a simple switch to enable OpenConfig Telemetry Support. By default, this feature is disabled and must be enabled.

1. Navigate to the device's *Configuration Terminal* context.

```

SLX#
SLX# config terminal
SLX (config)#

```

2. Use the `operational-state syncup enable <module>` command to enable all the supported modules or a specific module.

The `operational-state syncup enable all` command enables support for all the modules.

Use the specific module switch to enable OpenConfig Telemetry for that module.

```

SLX (config)# operational-state syncup enable ?
Possible completions:
  All          Enable oper db syncup for all modules
  Bgp          Enable oper db syncup for bgp
  Interface    Enable oper db syncup for interface
  Platform     Enable platform specific oper db syncup

SLX (config)#

```

The following example enables OpenConfig Telemetry Support for the BGP module.

```

SLX (config)# operational-state syncup enable bgp

```

```
SLX (config)#
```

Securing OpenConfig Telemetry Connections

This topic describes the steps to secure incoming connections from gNMI clients. By default, the gNMI server on SLX-OS listens on the insecure port 9339. To secure incoming connections, you must configure a port (range 1024-49151) on which the gNMI server listens on for incoming connections. The existence of this port configuration determines whether the gNMI server is listening for incoming connections in the secure or insecure mode.

It is assumed that your infrastructure is set up with a gNMI client. Its configuration is beyond the scope of this document.

Configuring a port for the gNMI server to listen on enables securing incoming connection.

1. Navigate to the device's *Configuration Terminal* context.

```
SLX#  
SLX# config terminal  
SLX (config)#
```

2. Navigate into the gNMI Server context.

```
SLX (config)# gnmi server  
SLX (config-gNMI-server)#
```

3. Configure a secure port on which to listen to incoming connections.

```
SLX (config-gNMI-server)# secure-port 48151  
SLX (config-gNMI-server)#
```

The gNMI server will start listening to incoming connections on port number 48151

Importing gNMI Server Private Key and Server Certificate

This topic describes the process to import the gNMI server's private key and server certificate in the *pkcs* format.

From within the SLX-OS context, run the **crypto ca import-pkcs** command and pass appropriate information for this command.

```
SLX # crypto ca import-pkcs type pkcs12 cert-type gNMI-server directory  
/root/certfolder/AV2 file av2.pfx host 10.x.x.x user root pass pass protocol SCP pkcs-  
passphrase pkcs
```

The server certificate is imported and saved.

Importing gNMI Client CA Root Certificate

This topic describes the steps to import the gNMI Client's CA root certificate in to the gNMI server. This enables password less connections between the client and the server.

It is assumed that your infrastructure is set up with a gNMI client. Its configuration is beyond the scope of this document.

Importing the gNMI client's CA Root Certificate for password less connection between the gNMI Server and Client.

From within the SLX-OS context, run the **crypto import gnmiclientca** command and pass appropriate information for this command.

```
SLX # crypto import gnmiclientca directory /root/certfolder/AV2/certs file ca.cert.pem  
host 10.x.x.x user root password pass protocol SCP
```

The CA certificate of the gNMI client is imported from the remote server.

Static Prefix Independent Convergence

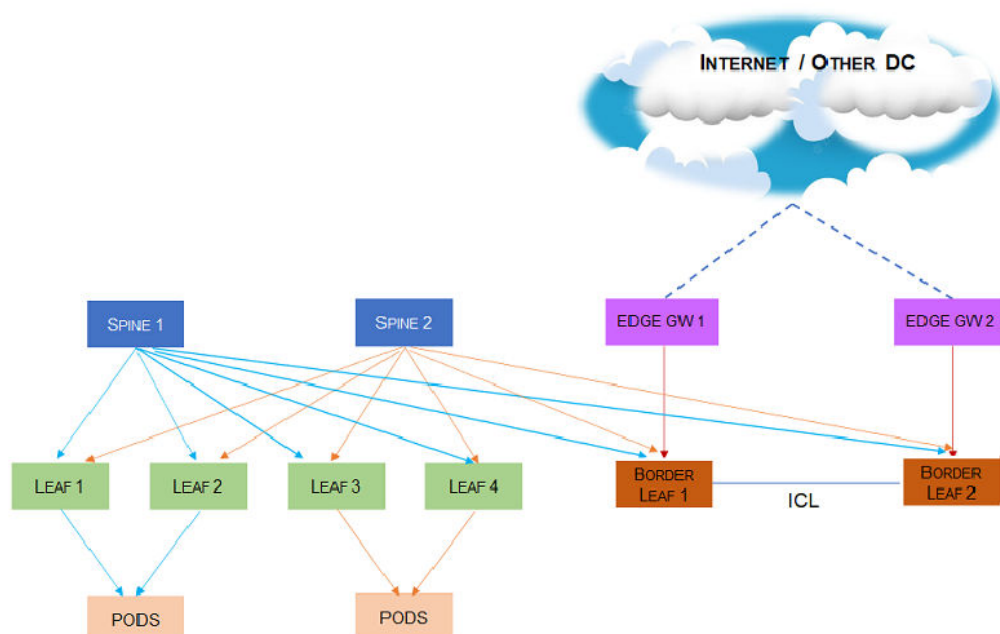
Static Prefix Independent Convergence

The time taken for a network to converge after any kind of link failure is an important factor in preventing major disruptions in the network. The Static Prefix Independent Convergence (Static PIC) feature reduces the time taken for the network to converge for static routes when there is a failover in primary static nexthop to backup static nexthop.

Static PIC is supported on all SLX platforms.

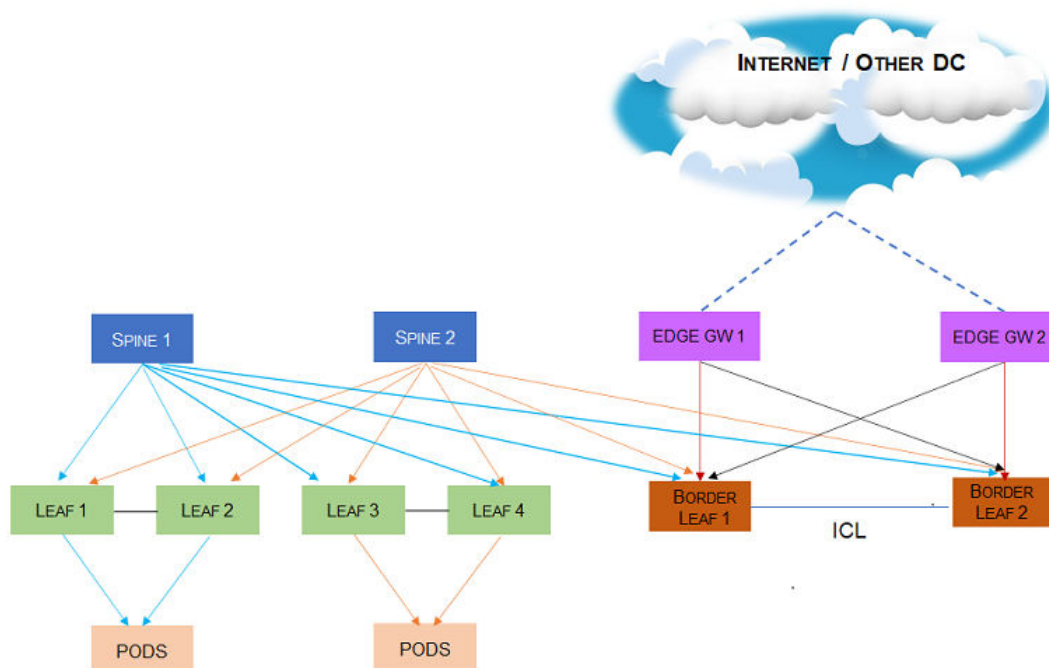
Single Gateway Failure

As seen in the following image, static routes are configured to reach a single Edge Gateway from each border leaf router. There is a redundant pathway for failover where the external network is connected through two Edge Gateways. In the case where the link between Border-Leaf 1 and Edge Gateway 1 fails, the traffic is automatically switched to the ICL link and then switched to the Edge Gateway 2.



Dual Gateway Failure

As seen in the following image, static routes are configured to reach the Edge Gateway from the border leaf routers per VRF. Generally, the link from a border leaf to the Edge Gateway is a port channel with ECMP links. When all ECMPs in the port channel towards both the Edge Gateways go down, the traffic is immediately switched to the ICL link and then switched to the other Border Leaf.



Configuring Static PIC

To configure Static Prefix Independent Convergence, do the following:

1. Navigate to the Configuration Mode.

```
SLX # configure terminal
SLX (config) #
```

2. Execute the `prefix-independent-convergence-static` command to enable Static PIC.

```
SLX (config)#prefix-independent-convergence-static
SLX (config)#
%Warning: Please run "clear ip[v6] route all vrf <vrf name>" for all static routes configured in VRFs
```

3. Exit out of the Configuration Mode.

```
SLX (config)# exit
SLX #
```

When configured, if there is a link fail, the traffic is switched over ICL to the configured backup Border Leaf.

This example configures Static PIC.

```
SLX (config)# prefix-independent-convergence-static
SLX (config)#
%Warning: Please run "clear ip[v6] route all vrf <vrf name>" for all static routes configured in VRFs
```




SLX-OS and Linux Shell Interoperability

[Overview](#) on page 97

[Launching Linux shell from SLX-OS](#) on page 98

[Executing Linux shell commands from SLX-OS](#) on page 99

[Executing scripts from SLX-OS](#) on page 100

[Accessing the Linux shell from SLX-OS](#) on page 101

[Executing SLX-OS commands from the Linux shell](#) on page 101

[Escalating Linux permissions to root](#) on page 102

[Saving and appending show command output to a file](#) on page 103

[Logs of Linux shell activities](#) on page 103

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*.

Launching Linux shell from SLX-OS

You can launch the Linux shell from within the SLX-OS CLI.

The Linux shell provides additional features such as running specific and permitted Linux commands and scripts to enable better and easier management of your SLX-OS device.

To launch the Linux shell:

1. From the Privilege Execution Mode, use the **start-shell** command to launch the Linux shell.

```
device# start-shell
```

- A disclaimer is immediately displayed. The content of this disclaimer may change from time to time.
- The privileges available within the Linux shell will be the same as the privilege as assigned to the SLX-OS user account used to launch the Linux shell.

```
device # start-shell
Disclaimer for Linux shell usage!

The Linux shell access on this SLX device is intended for diagnostics and
debugging purposes solely by the equipment vendor's trained engineers.
Improper use of the functionality made available through Linux shell
access could cause significant harm and disruption to the network operation.

Your use of the functionality made available through Linux shell access
is at your sole risk and you assume all liability resulting from such use.
The equipment vendor shall have no liability for any losses or damages
arising from or relating to Linux shell access (and the functionality
enabled thereby) by anyone other than the equipment vendor's
authorized engineers.

Proceeding with the usage of Linux shell access on this device explicitly
indicates your agreement to the terms of this disclaimer.
```

```
Entering Linux shell for the user: admin
[admin@device]#
```

The Linux shell is now ready to use.

2. To exit out of the Linux shell, use the **exit** command.

```
[admin@device]# exit
exit

Exited from Linux shell

device#
```

The Linux shell session is closed and you are returned to the SLX-OS Privilege Execution Mode prompt.

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.

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

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

6. 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.

1. 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.

2. 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
```

3. 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.

4. 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 106

[Insight Interface and TPVM](#) on page 111

[TPVM](#) on page 127

[TPVM Configuration Persistence](#) on page 150

TPVM, or Third-Party Virtual Machine, is a general server that resides on Extreme Networks SLX-OS devices. The guest OS that it provides is different from SLX-OS. TPVM supports both IPv4 and IPv6 address assignments.

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 TVPM in Baremetal mode on these devices - SLX 9540, SLX 9640, SLX 9150, and SLX 9250. The figure shows the baremetal mode and how multiple operating systems are stacked. The table following the figure provides the platform support details.

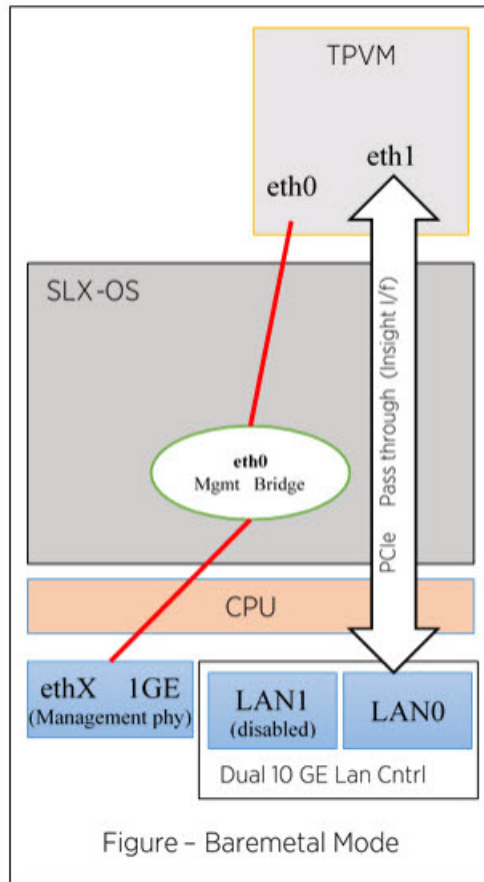


Figure 7: SLX-OS VM and Baremetal Modes

Table 14: Platform OS Mode Details

Platform	OS Mode	Host Image Version	OS	SLX-OS Image Version	TPVM Image Version	Insight Interface
SLX 9540	Baremetal	N/A		Linux SLX 5.3 HWE	Ubuntu 18.04.4 LTS	Yes
SLX 9640	Baremetal	N/A		Linux SLX 5.3 HWE	Ubuntu 18.04.4 LTS	Yes
SLX 9150	Baremetal	N/A		Linux SLX 5.3 HWE	Ubuntu 18.04.4 LTS	Yes
SLX 9250	Baremetal	N/A		Linux SLX 5.3 HWE	Ubuntu 18.04.4 LTS	Yes
8720-32C	Baremetal	N/A		Linux SLX 5.3 HWE	Ubuntu 18.04.4. 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 will assign the following ports for VM access after the port-channel has been configured using the `insight enable` command:

- 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 15: 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	
TPVM	extreme	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 the 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.

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.

```
The tpvm console command 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 18.04.4 LTS TPVM ttyS0
TPVM login:
```

Once in the TPVM console, you can execute `ctrl+\` on the SLX-9450, to switch back to the session from where the TPVM console was started.

Once in the TPVM console, you can execute `ctrl+\` to switch back to the session from where the TPVM console was started.

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#
```

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 Console Access

The **tpvm console** command 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 18.04.4 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.

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 login. Use *'extreme'* as the user name and *'password'* as the password. You can then use standard Ubuntu (18.04 LTS) commands to configure the IP address of the *'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 configured through 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 TVPM, 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)

Table 16: Insight Interface Support on Extreme SLX platforms (continued)

Extreme SLX device	Support for Insight Interface
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 cannot 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.

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 is 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 the 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.

All platforms are in baremetal mode in SLX-OS 20.2.x. A new command **tpvm console** allows connection to the TPVM console directly from an SLX *Telnet* or *Console* session.

The following is an example of logging to a TPVM console using the **tpvm console** command.

```
[SLX]# tpvm console
Connected to domain TPVM
Escape character is ^\
Ubuntu 18.04.4 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.

Configuring the Insight Interface for the SLX 9150/9250

The SLX 9150 and SLX 9250 use directly connected, dedicated internal ethernet ports for TPVM access. These ports are used exclusively for VM access and are not accessible

from the command line interface. Use the following steps to configure a port channel to access the Insight Interface on the SLX 9150 and SLX 9250.

1. Enter global configuration mode.

```
device# configure terminal
```

2. 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 50
```

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

```
device(config-Port-channel-50)# insight enable
```

4. Set the port ip-address.

```
device(config-Port-channel-50)# ip address 10.0.0.1/24
```

5. Enable the interface.

```
device(config-Port-channel-50)# no shutdown
```

6. 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 50
Port-channel 50 is up, line protocol is up
Insight mode is enabled
Hardware is AGGREGATE, address is f46e.959f.1af5
  Current address is f46e.959f.1af5
Interface index (ifindex) is 671088690 (0x28000032)
Minimum number of links to bring Port-channel up is 1
MTU 9216 bytes
LineSpeed Actual      : 10000 Mbit
Allowed Member Speed : 10000 Mbit
Priority Tag disable
Forward LACP PDU: Disable
Route Only: Disabled
Tag-type: 0x8100
Last clearing of show interface counters: 4d18h22m
Queueing strategy: fifo
FEC Mode - Disabled
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:
  12 packets, 5589 bytes
  Unicasts: 0, Multicasts: 12, 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: 00:00:25
```

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.

- Exit to global configuration mode.

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

- 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
```

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

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

- Set the port ip-address.

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

- Enable the interface.

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

- 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 119

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 mac1 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 mac1 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 mac1
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 mac1 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 122.

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.

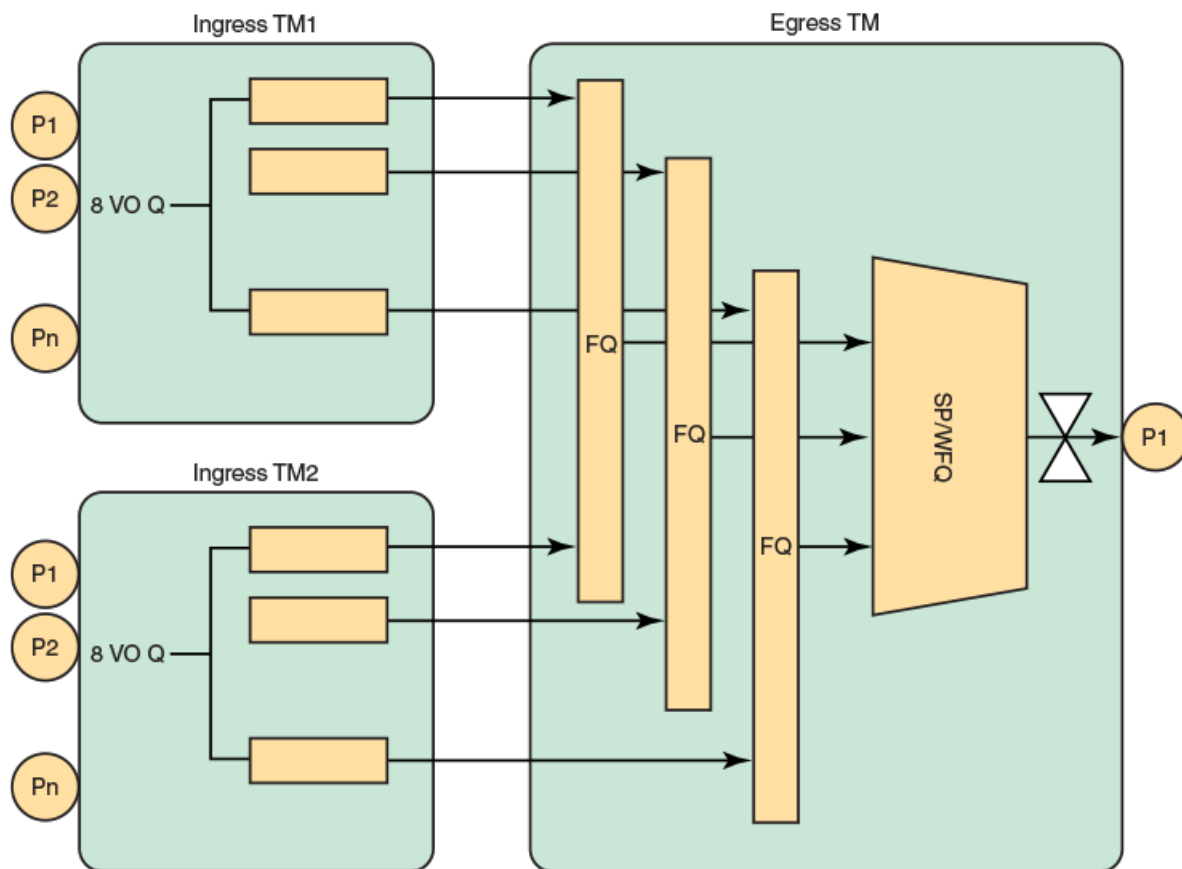


Figure 8: QoS egress scheduling scheme

In the figure above, P1 is the insight interface.

See the topic [Configuring QoS egress scheduling](#) on page 122 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
  Runt: 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
  Runt: 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
- Tcpdump: 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 (Iperf, 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 Management

This section covers TPVM management details.

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 5.3 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 113.



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 148.

TPVM and NTP

On each **tpvm start** event, TPVM will synchronize the clock with SLX. By default, TPVM will sync with the Ubuntu time server. You can use **show tpvm config ntp** to verify the NTP server, status, time, timezone, and clock settings. If necessary, **tpvm config ntp** can be used to configure a different NTP server.

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 Tcpcdump, 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.

TPVM Package Information

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.



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 TVPM 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-debian-file-name>.deb
```

- See the *Extreme SLX-OS Release Notes* for this SLX-OS version for information about the supported TPVM version.



Note

The TPVM Debian installation file will be deleted post successful deploy or upgrade of TPVM. This is to save hard disk space on the device.

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:

```
class="+ topic/pre pr-d/codeblock ">tpvm install
device# tpvm install
```

Uninstall TPVM:

```
class="+ topic/pre pr-d/codeblock ">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 the *tpvm config* Command

The **tpvm config** command is available at the SLX-OS CLI on a device.

The **tpvm config** command, in privileged EXEC mode, allows you to customize the following TPVM optional features:

- **tpvm config ldap**: LDAP configurations for TPVM
- **tpvm config dns**: DNS configurations for TPVM
- **tpvm config ntp**: NTP configurations for TPVM

tpvm config ldap

This section describes the commands used to configure LDAP over TPVM.

- **add**: Add LDAP configurations
 - **basedn**: Base Domain Name
 - **host**: LDAP server IPV4/IPV6 address or FQDN

- **rootdn**: Root Domain Name
- **rootdnpw**: Password for Root Domain Name
- **ca-cert**: LDAP TLS certificate operations
 - **import**: Import certificates for LDAP over TLS.
 - **directory**: Remote Directory
 - **filename**: Certificate filename
 - **host**: Hostname/IP address
 - **password**: Password for the user
 - **protocol**: Protocol (SCP)
 - **user**: Login name in the host
 - **remove**: Remove certificates for LDAP over TLS
- **remove**: Remove LDAP configurations
 - **basedn**: Base Domain Name
 - **host**: LDAP server IPV4/IPV6 address or FQDN
 - **rootdn**: Root Domain Name
 - **rootdnpw**: Password for Root Domain Name

tpvm config dns

This section describes the commands used to configure DNS over TPVM.

- **add**: Add DNS configuration
 - **dns-server**: Name Server List <IPv4 address> domain-name
- **remove**: Remove all DNS configurations

tpvm config ntp

This section describes the commands used to configure NTP over TPVM.

- **add**: Add NTP configurations
 - **server**: NTP server IPV4 address or FQDN
- **default**: Reset NTP configurations to default
- **remove**: Remove NTP configurations
 - **server**: NTP server IPV4 address or FQDN

Show Commands

This section describes the show commands used with **tpvm config**.

- **show tpvm config ldap**: Show TPVM LDAP server configuration
- **show tpvm config dns**: Show TPVM DNS server configuration
- **show tpvm config ntp**: Show TPVM NTP server configuration

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 administrator-level account
- Set TPVM autoboot
- Start or boot the TPVM



Note

SLX 20.1.2 and later releases have the flexibility to run either `tpvm-3.0.0` or `tpvm-4.0.0`. The default administrator passwords are different for each version. `tpvm-3.0.0` uses `admin/password`, and `tpvm-4.0.0` uses `extreme/password`. SLX 9540 is upgraded from hypervisor mode to bare metal mode in 20.2.x.

Prerequisites and considerations when using the `tpvm deploy` command:

- Extracted TPVM Debian package image – available in the `/tftpbboot/SWBD2900` folder. If TPVM is already installed, skip this step.
- An Advanced Features License. Use the following command to activate the license:
`license eula accept ADVANCED_FEATURES`.
- SLX 20.1.2 and later releases have the flexibility to run either `tpvm-3.0.0` or `tpvm-4.0.0`. Either one of the TPVM Debian package images can be copied to `/tftpbboot/SWBD2900` and installed using `tpvm install` or `tpvm deploy`. However, it is recommended that only one TPVM Debian package image (the desired version `tpvm`) exists under `/tftpbboot/SWBD2900` before executing `tpvm install` or `tpvm deploy`.

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

TPVM has two ethernet interfaces `eth0` and `eth1`. When using **`tpvm install`** the default setting is `eth0` and DHCP. When using the **`tpvm deploy`**, the user can specify

the interface to use `eth0` or `eth1` and set it to use either DHCP or specify an IP/netmask and gateway. In either case, the unused interface is set to manual mode without any IP configuration.

**Note**

The user can manually configure or override these settings by logging into the TPVM via **ssh** or **tpvm console** and manually modifying the linux network configuration.

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" extreme@10.23.30.153
```

When using the `passwordless` parameter, 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 that `sudo` for this user does not ask for a password. Once set, this parameter persists for the lifetime of the TPVM.

If not set, the default behavior requires a password for `sudo` activities, as dictated by the Ubuntu 18.04 LTS Server Operating System.

TPVM Password

The `tpvm-4.0.0` package ships with SLX-OS 20.1.2 and uses `extreme/password` as the default login credential. To automate the TPVM setup and achieve one touch provisioning of TPVM, this optional parameter sets the password for the TPVM admin user account. Once set, this parameter persists for the lifetime of the TPVM.

**Note**

The `tpvm-3.0.0` package can be used/installed with SLX-OS 20.1.2. If you are using this TPVM package, note that the default login credentials are `admin/password`.

TPVM Auto-boot

This option restarts the TPVM image automatically in subsequent reboots, such as an SLX-OS start on a Baremetal platform.

TPVM Start

After configuring the TPVM, `tpvm deploy` starts TPVM. A reboot of the SLX-OS also reboots the TPVM.

Upgrading TPVM

The installed instance of TPVM can be upgraded fully or incrementally when upgrading/downgrading TPVM. When upgrading/downgrading TPVM, it was required that TPVM be re-installed completely and its configuration was later restored from configuration persisted during the upgrade process. The general full upgrade process was to download the TPVM file to the SLX device. Once the download is successful, the existing TPVM instance was stopped and a snapshot taken. Post this, the TPVM was uninstalled. The new version of TPVM was then installed and its configuration restored from stored configuration.

From SLXOS version 20.4.1 onwards, support for incremental upgrade is available. When incrementally upgrading TPVM, the TPVM image is downloaded, along with the OS packages, updated scripts, and XML files. The existing TPVM installation is stopped, and the OS packages are updated first, followed by replacement of existing scripts and XML files. The last upgrade step is to upgrade the TPVM itself.

From SLX-OS version 20.6.3b, it is now possible to force the user to change the default password for the TPVM user 'extreme'. Keeping the default password which is well known, is a security risk. For more information, see [Force change of default password](#) on page 141.

When incrementally upgrading TPVM, the following requirements must be met:

- A minimum of 1Gb space must be available for TPVM upgrade.
- The existing installation of TPVM must be running for incremental upgrade to work.
- The minimum required TPVM version is 4.5.0 and the minimum SLXOS version is 20.4.1. If these constraints are not met, you must perform full installation of TPVM.

When TPVM upgrade fails, the following actions are performed to restore the previous installation:

- The old Scripts and XML files are restored from backup.
- The TPVM instance is restored from the snapshot (if available and only for full installations). For incremental upgrade, the updated OS files are replaced with the older copies.
- Log entries are created with the failure reasons.

This example shows the command to perform an incremental upgrade of TPVM.

```
SLX# tpvm upgrade incremental directory /proj/tpvm_upgrade/ filename
tpvm_inc_upg-4.5.0-0.amd64.deb host 10.10.10.1 user fav password testpassword protocol
scp use-vrf mgmt-vrf
```

This example shows the command to perform a full installation of TPVM.

```
SLX# tpvm upgrade directory /proj/tpvm_upgrade/ filename tpvm-4.5.0-0.amd64.deb host
10.10.10.1 user fav password testpassword protocol scp use-vrf mgmt-vrf
```

Uninstalling TPVM completely

The **undeploy-force** command is used to undeploy the TPVM and delete the disks.

Command Overview

The **undeploy-force** command performs the following operations:

- Uninstallation of TPVM
- Deletion of disks associated with TPVM
- Enable the uninstallation of TPVM in configuration mode

TPVM User Management

Describes the process to manage TPVM user account passwords.

The default password for the TPVM user account is well known. If used as such it can pose a security threat in any environment.

A two pronged approach is required to ensure that the default password is detected and the user provided an opportunity to change it to a strong password.

First, detect that the default password is used for the TPVM user account and inform the user about it. Next, enforce a set of password constraints to strengthen the password when updating or creating the password.

For more information on detecting default password, see [Force change of default password](#) on page 141.

For more information on configuring password length and strength, see [Strong TPVM password](#) on page 142.

Force change of default password

Describes the steps to force the change of the default password for the TPVM account 'extreme'.

The default password for TPVM user account *extreme* is well known, and, when not changed, is a security risk in any environment. It is recommended to change the password as soon as the TPVM is installed or upgraded.

From SLX-OS version 20.6.3b onwards, it is possible to check if an existing TPVM instance is retaining the default password and, when found, warn the user or abort the TPVM upgrade.

You can also enforce changing of the default password as the TPVM is upgraded. Use the *force-default-password-change* option of the **tpvm upgrade** command to enforce this change.

The TPVM upgrade process, when the *force-default-password-change* parameter is passed, checks if a password is configured for the TPVM user in an existing installation of TPVM. If the configured password is the default password, then the process aborts and alerts the user to create a password for the existing TPVM instance. Console log, DCM-1458, is generated.

Checks for default password happen in the following scenarios. The check process warns the user if it encounters the default password during the following activities:

- during SLX-OS upgrade, post reboot. This warning is generated for an existing deployment of TPVM.
- during reboot of the SLX-OS device, during the configuration replay step.
- during the process of installing a fresh TPVM instance.

Console logs, DCM-1457, are generated when default TPVM password is encountered.

This example shows the command to force TPVM default password update during an incremental update.

```
SLX# tpvm upgrade incremental directory /proj/tpvm_upgrade/ filename
tpvm_inc_upg-4.5.0-0.amd64.deb host 10.10.10.1 user fav password testpassword protocol
scp use-vrf mgmt-vrf force-default-password-change
```

Strong TPVM password

Describes how to strengthen the TPVM default user account's password.

Strong passwords are essential to maintain security for any system. TPVM is a part of the SLX-OS and securing its user password strengthens the security of SLX-OS itself.



Note

You can change the tpvm user password while tpvm is running. Legacy method of changing TPVM password is no longer supported, since the authentication of TPVM password now happens within the running TPVM instance. It is no longer required to change the password when TPVM is stopped (it is in the installed state).

Will not affect previously set passwords for any TPVM user.

As part of strengthening the TPVM user account, the TPVM user password must confirm to a set of constraints as to its length and the number of different components.

The following constraints are imposed on the TPVM user password:

- length - The password must conform to be of a minimum length which is configurable. The password, at the minimum, must be eight (8) characters long and can be up to forty (40) characters long.
- upper case characters - The password must conform to having a configurable minimum number of upper case characters.
- lower case characters - The password must conform to having a configurable minimum number of lower case characters.
- numbers - The password must conform to having a configurable minimum number of digits.
- special characters - The password must conform to having a configurable minimum number of special characters.

The following special characters are supported:

```
[ @ # $ % ^ * ( ) _ + - = { } [ ] | : ; " < > ? / ~ ` . , ] *
```

The following special characters are not supported

```
& ! \ ' "
```

TPVM user password strength and length is configured from the **password-policy** command within the TPVM config mode.

This example shows the configuration of the TPVM user password's minimum length to 14 characters.

```
SLX# configure terminal
SLX (config)# tpvm TPVM
SLX (config-tpvm-TPVM)# password-policy min-length 14
SLX (config-tpvm-TPVM)#
```

This example shows the configuration of the required character content of the TPVM user password.

```
SLX# configure terminal
SLX (config)# tpvm TPVM
SLX (config-tpvm-TPVM)# password-policy lower 8
SLX (config-tpvm-TPVM)# password-policy upper 2
SLX (config-tpvm-TPVM)# password-policy number 2
SLX (config-tpvm-TPVM)# password-policy special-char 2
SLX (config-tpvm-TPVM)# exit
SLX (config)# exit
SLX# show running-config tpvm
tpvm TPVM
...

password-policy min-length 14
password-policy character-restriction lower 8
password-policy character-restriction upper 2
password-policy character-restriction numeric 2
password-policy character-restriction special-char 2
SLX#
```

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 58118E89F3A912897C070ADB76221572C52609D**

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 17: 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 18: cURL commands

Command	Description
sudo apt-get update	Updates any dependency packages
sudo apt-get install curl	Installs the latest cURL package, or specifies a previous version number
sudo apt-get upgrade curl	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 19: Google-chrome commands

Command	Description
sudo apt-get update	Updates any dependency packages
sudo apt-get install google-chrome-stable	Installs the latest Google-chrome package, or specifies a previous version number
sudo apt-get upgrade google-chrome-stable	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 20: ifconfig and route command options

Command	Description
ifconfig eth0 <i>Net-IP-Addr</i> netmask <i><Net-IP-Addr-Mask></i>	Checks interface status and statistics
route add -net <i>Net-IP-Addr</i> netmask <i>Net-IP-Addr-Mask</i> gw <i>Gw-IP</i>	Adds a route
route add default gw <i>GW-IP</i>	Adds a default gateway

Ethtool

The Ethtool utility is used to get device information. The following table lists useful command options.

Table 21: Ethtool commands

Command	Description
sudo apt-get update	Updates any dependency packages
sudo apt-get install ethtool	Installs the latest Ethtool package, or specifies a previous version number
sudo apt-get upgrade ethtool	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 22: 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 23: 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 24: 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
```

Update TPVM Gateway

Changes to the gateway address for the TPVM management interface, required you to bring down the TPVM, make the change, and then bring up the TPVM again.

From SLX-OS version 20.3.2d onwards, changes to the gateway for TPVM management interface does not require you to bring down the TPVM. These changes are applied dynamically and does not require TPVM reboot.



Caution

Updating the TPVM gateway with incorrect gateway address might result in disconnection from NTP, DNS, LDAP, and other services.

IPv6 Support for TPVM

TPVM supports configuring its management interfaces and various other services such as NTP, DNS, LDAP, etc. using IPv6.

The following commands now support configuring using IPv6.

- **ntp**
- **dns**
- **interface management**
- **tpvm upgrade**
- **trusted-peer**
- **tpvm download**
- **ldap ca-cert**
- **ldap host**

Downgrading TPVM to a lower version when IPv6 is configured

Care should be taken to remove the IPv6 configuration before downgrading to a lower TPVM version. Any configuration that uses IPv6 must be removed and reconfigured using IPv4.

When downgrading from SLX-OS 20.4.3 to SLX-OS 20.4.2, ensure that the IPv6 LDAP configurations are reset to using IPv4 before starting the downgrade. When downgrading to SLX-OS 20.4.1, ensure that any IPv6 configuration is reset to using IPv4 before starting the downgrade.

When downgrading from SLX-OS 20.4.3 to SLX-OS 20.3.x or below, ensure that no IPv6 configuration is present. All TPVM configurations must use IPv4 only.

TPVM Configuration Persistence

TPVM (Guest OS) instance on Bare-Metal SLX, also can be deployed using the *config tpvm mode*.

All the TPVM related configurations, including *config deploy* (this process installs and then starts a TPVM instance) is persisted in the SLX database and becomes a part of the SLX **running-configuration**. These persisted configuration values are then used while upgrading the installed TPVM or while RMA of the SLX device. The persisted TPVM configurations are not applied when the TPVM reboots due to reboot or power-cycle of the SLX-OS. Since these configurations are a part of the saved running configuration, they are automatically applied for the above events.

The ability to upgrade an existing TPVM instance and to create and manage Snapshots of the running TPVM instance is also available with TPVM Configuration Persistence.

The TPVM **snapshot** feature allows you to take the snapshot of the TPVM instance when the TPVM is not running (STOP state). Only one snapshot can be taken at any point of time. The TPVM snapshot can be used later to revert any failure in installing TPVM or to reinstall an old version of TPVM. The snapshot image is stored at the on the SLX device folder **/support/snapshots/TPVM/auto**.

When upgrading an existing TPVM instance, the new TPVM image is first downloaded and installed. If TPVM configuration is persisted, it is applied on the installed TPVM. As a backup measure, a snapshot of the previous TPVM installation is taken and stored on the SLX device. The new TPVM image (*debian package file*) is downloaded from the following local folder **/tftpboot/SWBD2900**.

For TPVM Configuration Persistence, the following configurations must be applied before executing the *config deploy*, as these three configurations are applied to the TPVM before it is deployed.

These configurations are:

1. **auto-boot** - The *auto-boot* configuration enables the TPVM to be auto-booted when the SLX device is power-cycled or rebooted for any reason.
2. **password** - The *password* configuration updates the password of default userid 'extreme' at TPVM instance, which has a SUDO role also. By default the value is *password*.
3. **interface management** - The *interface management* configuration updates the network interface *eth0* at TPVM instance used for management purposes. By default it is set to *dhcp*.

How it Works

How it works – TPVM Persistence

When using the Legacy method, the TPVM installation as well its configuration were configured through the 'Exec' mode commands. The *TPVM/Ubuntu* was an independent Guest OS with its own *RootFS* and it will make a copy or a static entry of the above executed configuration files. For example, if the **TPVM Management Interface 'eth0'** is configured, from SLX, the *TPVM Rootfs /etc/network/interface* file will be edited to maintain the static entry for its future use.

The above model is good for any kind of TPVM reboot or powercycle of the SLX device and is insufficient, if TPVM is upgraded with a new image or SLX switch. The RMA is also installed with the new SLX switch, but with the old SLX switch running-configurations.

In the new way of installing TPVM, installing TPVM and its configuration is supported in the '*Config*' mode. Whereas all the TPVM configurations are also persisted at the SLX OS Configuration database, so they become part of the switch 'running-configurations'.

**Note**

It can be either of the configuration modes, only one TPVM instance (**named TPVM**) is deployed or installed and its Debian image is expected to be available at the */tftpboot/SWBD2900* folder. The only major difference is being persistent at the SLX-OS, in the new *Config* mode.

The TPVM configurations are listed below as 3 types:

- Install time configuration
 - Password
 - Interface Management
 - Interface Insight (to be implemented in the future release).
- Run time configuration
 - auto-boot
 - hostname
 - timezone
 - dns
 - ntp
 - ldap host
 - ldap ca-cert
 - trusted-peer
 - disk (to be implemented in future release)
- Action
 - deploy

The **install time configuration** should be always configured before the **deploy** configuration, else they will fail. As they are applied to the TPVM Instance RootFS during the installation time. The settings before the **deploy** configuration persist, as these configurations are present in the SLX-OS running-configuration database. Whenever the **deploy** is configured, at that time, these configurations are applied during the TPVM installation stage.

The post **deploy** configuration, if required, will stop the TPVM, by the command **tpvm stop**. Then the *tpvm* config submode is set at the **install time configuration**, followed by the **tpvm start** command.

The **Run time configurations** may be configured any time. If configured before '**deploy**', they are just persisted at SLX-OS running-configuration Database. If configured after

'**deploy**' configuration, it implies that the TPVM is running and the configuration is first applied to the TPVM run-time. If they succeed only, then it is persisted to the SLX-OS running-configuration database.

The **Deploy** configuration implies the following:

- It will look for TPVM Debian image package file under */tftpboot/SWBD2900* folder.
- It will install the Debian image package.
 - ▪ poll for some timeout for install completion else report timeout
- It will apply the *Install time configuration* if found persisted in the SLX-OS running-configuration database.
- It will start the TPVM Guest OS.
 - The poll for timeout of the start completion, of the TPVM boot up, reports timeout. (The boot operation may still go on in the background).
- It will apply the **Run time configuration** if found persisted in the SLX-OS running-configuration database.

Across normal SLX-OS reboot, there is no change in the configuration especially in the *deploy* configuration. Post boot action is needed and the TPVM will reboot with its configurations and if *auto-boot* is set manually through the *tpvm start* command.



Note

The *interface insight* and the *disk* configurations, are currently not supported by the SLX-OS backend and it will report failure if tried configuring. By default for the first time the disk *vdb* is created and mounted to the */apps* TPVM legacy. The subsequent *deploy* or *install* will reuse the existing disk *vdb*. Commands like "**tpvm uninstall force**" is needed to purge the *vdb*.

How it works – across RMA

When the SLX switch is RMA and its replaced by another SLX switch, it's expected that the new switch will have the desired SLX-OS image as well as the TPVM Debian image file.

The *default-config* new switch will not have any TPVM related configurations. The corresponding cleanup will be needed, if any switch boots with the *default-config*.

The *Admin user* will have the backup of the *running-configuration* last switch, and it will copy the same to the new switch *running-configuration* through the command:

For example - *SLX# copy flash://old.cfg running-config*.

or

SLX# copy scp://<username>:<password>@hostname/<filepath> running-config

If copied, the *running-configuration* will have the *tpvm* configurations and they will be applied.

How it works – across upgrade

The new *tpvm upgrade* command can be explained as 2 main steps:

First download the new Debian image package file and then as explained above, re-deploy the TPVM.

Re-deploying means *stop* and *uninstall* running TPVM instances. Hence it's like fresh deploying.

As explained for deploying, if any TPVM configurations are found in the SLX-OS running-configuration Database, they are also applied to TPVM.

Deploy with TPVM Configuration Persistence

This topic describes how to *install* and *start* TPVM using the **deploy config** command and persist other TPVM configurations in the SLX-OS *running-config*. These persisted configurations are useful during the *TPVM upgrade* operation. Also, for SLX switch RMA use case, TPVM can be deployed with previous configurations, as per the *SLX RMA* workflow of restoring SLX configurations.

Pre-requisite for deploy:

The TPVM debian image file (e.g. *tpvm-4.2.5-1.amd64.deb*) is available in the following folder */tftpboot/SWBD2900*. To deploy, install this file with this naming convention.



Note

Only one file can be stored in the above mentioned location.

Example 1) This example describes a simple **deploy** action with TPVM built-in defaults.

```
SLX # config terminal
SLX (config)# tpvm TPVM
SLX (config-tpvm-TPVM) # deploy
SLX (config-tpvm-TPVM) # end
```



Note

- The *deploy* command is a time-taking operation. If executed through the *Netconf/RPC*, it is non-blocking to the client and will proceed in the background at the SLX switch. The *Raslog* “DCMD-1451” (for start), “DCMD-1452” (for success) and “DCMD-1453” (for fail/abort) are published.
- TPVM image significant defaults are:
 - *hostname* – ‘tpvm’
 - *timezone* – ‘Etc/GMT’
 - *management* - ‘dhcp’
 - *password for user ‘extreme’* – ‘password’
 - *dns* -
 - *ntp* – no setting
 - *ldap* – no setting
 - *trusted-peer* – not set

Example 2) This example describes **deploy** TPVM with *install time* configuration. The TPVM *config password*, *interface management* can be set only before 'starting' or 'deploy' of the TPVM.

```
SLX # config terminal
SLX (config)# tpvm TPVM
SLX (config-tpvm-TPVM) # password newpassword
SLX (config-tpvm-TPVM) # interface management ip 10.25.24.21/24
SLX (config-tpvm-TPVM) # deploy
```

Example 3) This example describes the *deploy* TPVM and later '*updates*' and '*any install time*' configurations.

```
SLX # config terminal
SLX (config)# tpvm TPVM
SLX (config-tpvm-TPVM) # password newpassword
SLX (config-tpvm-TPVM) # interface management ip 10.25.24.21/24
SLX (config-tpvm-TPVM) # deploy
SLX (config-tpvm-TPVM) # end
SLX # tpvm stop
SLX # config terminal
SLX (config)# tpvm TPVM
SLX (config-tpvm-TPVM) # password newpasswordagain
SLX (config-tpvm-TPVM) # interface management ip 10.25.24.221/24
SLX (config-tpvm-TPVM) # end
SLX # tpvm start
```

Example 4) This example describes the *config* TPVM with *any install time* or *runtime* configuration without *deploy*. The configuration here is just persisted in the SLX *running-config* database. The Runtime configuration is applied anytime before or after deploy. In the later case, the TPVM will be in the running status and configuration is applied to the TPVM instance first. Only upon success, the configuration will be persisted at the SLX-OS database.

```
SLX # config terminal
SLX (config)# tpvm TPVM
SLX (config-tpvm-TPVM) # password newpassword
SLX (config-tpvm-TPVM) # interface management ip 10.25.24.21/24
SLX (config-tpvm-TPVM) # auto-boot
SLX (config-tpvm-TPVM) # hostname newhostname
SLX (config-tpvm-TPVM) # timezone Europe/Stockholm
SLX (config-tpvm-TPVM) # end
```

Later TPVM may be installed and started with the above configuration by *config deploy*.

```
SLX # config terminal
SLX (config)# tpvm TPVM
SLX (config-tpvm-TPVM) # deploy
```

Example 5) This example describes the *deploy* TPVM with some configuration and later update any runtime configuration.

```
SLX # config terminal
SLX (config)# tpvm TPVM
SLX (config-tpvm-TPVM) # password newpassword
SLX (config-tpvm-TPVM) # interface management ip 10.25.24.21/24
SLX (config-tpvm-TPVM) # auto-boot
SLX (config-tpvm-TPVM) # hostname newhostname
SLX (config-tpvm-TPVM) # timezone Europe/Stockholm
SLX (config-tpvm-TPVM) # deploy
SLX (config-tpvm-TPVM) # end
```

```
SLX # config terminal
SLX (config)# tpvm TPVM
SLX (config-tpvm-TPVM) # hostname oldhostname
SLX (config-tpvm-TPVM) # no timezone
```

Example 6) This example describes the *set password* configuration using the encrypted string. Here the *newpassword* encrypted string is used.

```
SLX (config-tpvm-TPVM) # password $6$M@T]Ip^<$ .kavoRt.uINQoghhcpDYd4elQg3BWWuJDdVSmGtZcM/
Ry6IRUYKkKtJ80PgTXKFEVnzWx4oTSPvHPRtvqXtGf/
```

Example 7) This example describes the *stop* and *uninstall* TPVM configurations and also by using the *no deploy* but *retain* configuration.

```
SLX # config terminal
SLX (config)# tpvm TPVM
SLX (config-tpvm-TPVM) # no deploy
SLX (config-tpvm-TPVM) # end
```

Example 8) This example describes the *stop*, *uninstall*, *no deploy* and *purge all* TPVM configurations.

```
SLX # config terminal
SLX (config)# no tpvm TPVM
SLX (config) # end
```

Example 9) This example describes how to reset any individual configuration, by using the *no* prefix command, and certain configurations may have parameters too.

```
SLX (config-tpvm-TPVM) # no hostname
```

Managing Snapshots

The installed TPVM *rootfs* only snapshot's (backup) can be taken manually or as part of TPVM *upgrade* CLI. If admin finds that the upgrade failed for any reason, the TPVM instance can be reverted to the backup instance.



Note

The in-between configurations should not be updated and only one *snapshot* instance is supported. Upon revert the old TPVM instance with last applied configurations will be restored and rebooted.

Example 1) This example describes the *stop snapshot* TPVM. It will *stop* the TPVM if running, and take the snapshot of the TPVM. The backup operation is used later to *restore / revert* this TPVM.

```
SLX# tpvm snapshot create
(or)
SLX# tpvm snapshot create tpvmid TPVM
```



Note

1. The current implementation of the TPVM configuration should not be edited before *restore / revert* operations. If edited, the TPVM configuration and the persisted values will differ.
2. The *snapshot* is not taken for the TPVM any additional disk such as *vdb (/apps)*. But only the TPVM image (RootFS) is taken.
3. Only one snapshot is kept.
4. It *stop* TPVM if running, it will not re-start the TPVM instance.

Example 2) This example describes the *snapshot* TPVM along with *upgrade* TPVM. After download of new image file for *upgrade*, It will collect the *snapshot* of any TPVM in running or installed state, then perform *deploy* of the new image, if SLX-OS running-configuration, previously has deployed TPVM.

```
SLX# tpvm upgrade protocol scp host 10.31.2.101 directory /folder/to/debian filename
tpvm-4.2.5-1.amd64.deb user mylogin password mypasswd snapshot
```

Example 3) This example describes the *show* configuration and if there is any previous existing TPVM *snapshot*.

```
SLX# show tpvm snapshot

Sample output:
Domain name: TPVM
Domain version: 4.2.4
Snapshot time: Mon Jun 21 19:50:58 GMT 2021
Snapshot size: 4.4G
```

Example 4) This example describes the *delete snapshot*, if any.

```
SLX# tpvm snapshot delete
```

Example 5) This example describes the *revert to restore snapshot* versions of TPVM.

```
SLX# tpvm snapshot revert
```



Note

The above example is like TPVM reboot operation. It can *stop* and *uninstall* currently running TPVM, if any. It replaces the backedup *snapshot* version of the TPVM and *restarts* the TPVM. The TPVM reboots with its own copy of configurations available at the *RootFS*. The SLX-OS *running-configuration* is not applied to the TPVM during revert. Hence do not update any configurations, if revert is planned and update it post revert.

Upgrading TPVM

The installed instance of TPVM can be upgraded fully or incrementally when upgrading/downgrading TPVM. When upgrading/downgrading TPVM, it was required that TPVM be re-installed completely and its configuration was later restored from configuration persisted during the upgrade process. The general full upgrade process was to download the TPVM file to the SLX device. Once the download is successful, the existing TPVM instance was stopped and a snapshot taken. Post this, the TPVM was uninstalled. The new version of TPVM was then installed and its configuration restored from stored configuration.

From SLXOS version 20.4.1 onwards, support for incremental upgrade is available. When incrementally upgrading TPVM, the TPVM image is downloaded, along with the OS packages, updated scripts, and XML files. The existing TPVM installation is stopped, and the OS packages are updated first, followed by replacement of existing scripts and XML files. The last upgrade step is to upgrade the TPVM itself.

From SLX-OS version 20.6.3b, it is now possible to force the user to change the default password for the TPVM user 'extreme'. Keeping the default password which is well known, is a security risk. For more information, see [Force change of default password](#) on page 141.

When incrementally upgrading TPVM, the following requirements must be met:

- A minimum of 1Gb space must be available for TPVM upgrade.
- The existing installation of TPVM must be running for incremental upgrade to work.
- The minimum required TPVM version is 4.5.0 and the minimum SLXOS version is 20.4.1. If these constraints are not met, you must perform full installation of TPVM.

When TPVM upgrade fails, the following actions are performed to restore the previous installation:

- The old Scripts and XML files are restored from backup.
- The TPVM instance is restored from the snapshot (if available and only for full installations). For incremental upgrade, the updated OS files are replaced with the older copies.
- Log entries are created with the failure reasons.

This example shows the command to perform an incremental upgrade of TPVM.

```
SLX# tpvm upgrade incremental directory /proj/tpvm_upgrade/ filename
tpvm_inc_upg-4.5.0-0.amd64.deb host 10.10.10.1 user fav password testpassword protocol
scp use-vrf mgmt-vrf
```

This example shows the command to perform a full installation of TPVM.

```
SLX# tpvm upgrade directory /proj/tpvm_upgrade/ filename tpvm-4.5.0-0.amd64.deb host
10.10.10.1 user fav password testpassword protocol scp use-vrf mgmt-vrf
```

TPVM Migration

With the introduction of the TPVM Configuration Persistence feature, the ability to persist TPVM configuration after TPVM upgrade was also introduced. The TPVM Migration feature now adds the ability to migrate the old configuration and store it automatically into SLX-OS *running configuration*.

This feature, introduced in SLX-OS version 20.3.2a, migrates some of the TPVM configurations made using the *Privilege Execution Mode* of the SLX-OS. These changes are now added to the SLX-OS's *running configuration* automatically.

This migration happens the first time the SLX device is upgraded to SLX-OS version 20.3.2a. This is a one time activity and this action will become obsolete in the next version.



Note

No user action is required for this migration to happen.

The following table lists the various TPVM configurations and their migration status.

Configuration	Migration State	Notes
tpvm auto-boot	Migrated	
tpvm disk	Not Migrated	Disk configuration is not supported in the configuration mode, and therefore, not migrated.
tpvm password	Migrated	Only the old password is migrated. This is due to the password being encrypted and stored and it is not possible to know if the password was changed during the migration.
tpvm config ntp	Migrated	
tpvm config dns	Migrated	
tpvm config ldap	Migrated	Secure LDAP require certificates. It is assumed that certificates are already downloaded and installed. Certificates are not validated during this migration. A notification will be sent to the user to reconfigure LDAP certificate settings.
tpvm config hostname	Migrated	
tpvm config timezone	Migrated	

Configuration	Migration State	Notes
<code>tpvm deploy <interface> allow-pwless</code>	Not Migrated	This is the new default configuration and is not migrated.
<code>tpvm deploy mgmt [dhcp static]</code>	Migrated	
<code>tpvm deploy insight</code>	Not Migrated	Insight interface configuration is not supported when configuring using the Privilege Execution Mode commands.
<code>tpvm config ldap ca-cert</code>	Not Migrated	
<code>tpvm config trusted-peer</code>	Not Migrated	All trusted-peer configurations are not migrated.

**Note**

You must manually configure those parameters that were not migrated automatically to ensure that your TPVM's configuration is the same post migration.

Remove TPVM on Copying Default Config as Startup Config

The new default behaviour of the **copy default-config startup-config** command is to reset the device to its default config and also reinstall and recreate the installed TPVM (if installed) once the device is rebooted.

There will be instances where you would like to reset the device to its default state without automatically reinstalling the TPVM. To facilitate this activity, use the **copy default-config startup-config remove-TPVM** command. When executed, this command will not reinstall the TPVM on device reboot after copying the default configuration as the startup configuration.

Uninstalling TPVM completely

The **no deploy** command brings down and removes the TPVM and associated disks from the current configuration. However, in some cases, the command might fail to remove the configured disks.

To completely remove TPVM and its associated disks, use the **undeploy-force** command. The command removes associated disks from the current configuration without fail. This command can also be used to forcibly remove any virtual disks that were retained when TPVM was uninstalled using the *no deploy* command.

Uninstalling TPVM completely

The **no deploy** command brings down and removes the TPVM and associated disks from the current configuration. However, in some cases, the command might fail to remove the configured disks.

To completely remove TPVM and its associated disks, use the **undeploy-force** command. The command removes associated disks from the current configuration without fail. This command can also be used to forcibly remove any virtual disks that were retained when TPVM was uninstalled using the *no deploy* command.

Restoring Configuration from Backup File

When using a file to restore a backed up configuration as the startup configuration, care must be taken to ensure that the source file's TPVM configuration is the same as the running TPVM's configuration. Particularly, if the source's TPVM configuration is partial, and does not have some of the configured parameters of the running TPVM instance. In such a scenario, it is advised not to proceed with this change as it will result in the configuration of the installed TPVM not being similar to the configuration of the previous TPVM instance.



Measured Boot

[Measured Boot](#) on page 162

[Verifying Measured Boot](#) on page 162

Measured Boot

Measured Boot is a mechanism to ensure that the integrity of the firmware and software running on a SLX hardware platform is maintained. This is ensured by the calculating a hash of the values of each stage in the boot process and comparing these values with the values stored on a remote verification server.

During device boot, before each stage of the boot process is executed, a measurement of the stage to be executed is calculated and extended (copied) to the *Trusted Platform Module* (TPM) chip. Once the device completes booting, these measurements are verified with a remote server and if an issue is discovered, the administrators of the device are notified.

Unlike the Secure Boot process, where the boot process is aborted when an error is encountered, the Measured Boot process allows the device to come up completely and alerts the device administrators of any issue encountered during the process.

The **measured-boot** command enables the Measured Boot feature on a SLX device. This feature is disabled by default.

Supported Platforms

This feature is supported on Extreme 8720 and Extreme 8520 devices.

Verifying Measured Boot

Measured Boot is a mechanism to ensure that the integrity of the firmware and software running on a SLX hardware platform is maintained. This is ensured by the calculating a hash of the values of each stage in the boot process and comparing these values with the values stored on a remote verification server.

The following commands shows you how to view the various certificates installed in the Trusted Platform Module (TPM) chip on the hardware device.

1. Use the **show tpm** command.

```
SLX# show tpm
Possible completions:
```

```
ekcert Show Endorsement Key (EK) certificate
iakcert Show Initial Attestation Key (IAK) certificate
idevidcert Show Initial Device Identifier (IDevID) certificate

SLX# show tpm ?
Possible completions:
ekcert      Show Endorsement Key (EK) certificate
iakcert     Show Initial Attestation Key (IAK) certificate
idevidcert  Show Initial Device Identifier (IDevID) certificate
```

2. Use the **show system** command to view the current provisioning state of the TPM chip.

```
SLX # show system
Stack MAC : 40:88:2f:c1:b4:1d
-- UNIT 0 --
Unit Name : SLX
Up Time : up 23 min
Current Time : 21:32:02 GMT
SLX-OS Version : 20.3.3
Jumbo Capable : yes
Burned In MAC : 40:88:2f:c1:b4:18
.
.
-- Fan Status --
Fan 1 is Ok, speed is 6400 RPM
Fan 2 is Ok, speed is 6400 RPM
Fan 3 is Ok, speed is 6560 RPM
Fan 4 is Ok, speed is 6400 RPM
Fan 5 is Ok, speed is 6400 RPM
Fan 6 is Ok, speed is 6400 RPM
-- TPM Status --
TPM Provision status: Provisioned
```



Remote Attestation

[Remote Attestation](#) on page 164

[Configure Remote Attestation](#) on page 165

Remote Attestation

The Remote Attestation feature enables each embedded network device (client) to authenticate its hardware and software components to a remote server (attestation server). The hardware and software components that can be attested are:

- the PCR register contents of the client device's TPM chip.
- hashes of chosen files/binaries/libraries from the client device.

This feature is supported on the Extreme 8720 and Extreme 8520 devices.

The primary function of remote attestation is to prevent offline tampering of the underlying firmware of the various hardware in the network. Since these measurements are done periodically and then updated to the Remote Attestation server, any offline changes are caught and flagged immediately.

This feature uses *Keylime*, an Open Source project for Remote Attestation as the remote attestation server. For more information see <https://keylime.dev>.



Note

Configuration of the Keylime Remote Attestation server is outside the scope of this document. Refer to its documentation.



Note

For the supported version of the Keylime Remote Attestation server, see the release notes for this SLX-OS software release.

Remote Attestation works by comparing the hashes generated by various network devices with a known hash. This known hash is generated at the time of building the SLX-OS and is published by Extreme Networks along with the SLX-OS software.

Allowlist is a file that contains a list of hash values of various components of the SLX-OS. This list is published as a MS-Excel workbook and a plain text file. Each release of SLX-OS will have its own Allowlist file that is released along with the SLX-OS software. If you want to implement Remote Attestation, you must download the copy of the *Allowlist* appropriate for your SLX-OS release and upload it to your Keylime server. This

file will then be used to compare the values sent by your network devices for Remote Attestation purposes.

Along with the *Allowedlist*, the hash of each boot component is published to the Keylime server, so that it can be compared against the PCR register content. This value is used to verify the integrity of the boot file by comparing with the value sent by the client device.

Remote Attestation uses Linux® *Integrity Measurement Architecture* (IMA). IMA maintains a runtime measurement(SHA256 hash) list for all or some selected files/binaries/ libraries on the SLX-OS. These measurements are compared against the *allowlist* to verify their integrity.

Keylime Components

The complete Remote Attestation infrastructure consists of the following components:

- **Keylime Agent:** This component measures the various hashes used for Remote Attestation. This component is installed on each SLX-OS device that needs Remote Attestation.
- **Keylime Verifier:** This component periodically verifies the integrity state of the SLX-OS device on which the Keylime Agent is running on. This component is installed on each SLX-OS device that needs Remote Attestation.
- **Keylime Tenant:** This component inputs the *Allowlist* file data to the Keylime Agent. This component is installed on each SLX-OS device on which the Keylime Agent is running on.
- **Keylime Registrar:** This component maintains a database of all the Keylime Agents registered with the Keylime Server.



Note

Refer to the Release Notes for the supported *Keylime* version.

Configure Remote Attestation

For Remote Attestation to work, Measured Boot must be enabled. To learn more about Measured Boot, see [Measured Boot](#) on page 162 in this document.



Note

Remote Attestation is supported on Extreme 8720 and Extreme 8520 devices.

To configure Remote Attestation do the following:

1. Navigate into the configuration mode

```
SLX # configure terminal
SLX (config)#
```

2. Enable Measured Boot.

```
SLX (config)# measure-boot enable
SLX (config)#
```

3. Enter the Remote Attestation configuration mode.

```
SLX (config)# remote-attestation
SLX (config-remote-attestation)#
```

4. Configure the remote Remote Attestation server first. Here you will configure the settings for the VRF that will be used to access the Remote Attestation server.

```
SLX (config-remote-attestation)# registrar-server 10.1.1.1 use-vrf default-vrf
SLX (config-remote-attestation-10.1.1.1/default-vrf)#
```

5. Configure the port on which the Remote Attestation server is listening. This configuration is done on the VRF configured in the previous step.

```
SLX (config-remote-attestation-10.1.1.1/default-vrf)# registrar-port
SLX (config-remote-attestation-10.1.1.1/default-vrf)#
```

The default port is 8890. If your Remote Attestation server is listening on another port, enter that port number here. Valid values are in the range 0-65535.

6. Exit out of *registrar-server* configuration mode.

```
SLX (config-remote-attestation-10.1.1.1/default-vrf)# exit
SLX (config-remote-attestation) #
```

7. Configure the Remote Attestation Agent UUID.

```
SLX(config-remote-attestation)# agent-uuid ?
SLX (config-remote-attestation) #
Possible completions:
<1-100>   UUID of the device.[auto]
```

The **show remote-attestation status** displays UUID status.

8. Configure the Remote Attestation listening port.

```
SLX(config-remote-attestation)# agent-port ?
SLX (config-remote-attestation) #
Possible completions:
<Choose the port number in the range 0 - 65535>
SLX(config-remote-attestation)# agent-port
```

The Remote Attestation agent is configured to listen on port 9002 by default. If your Remote Attestation agent is listening on a different port, configure that port here. Port can be in the range 0-65535.

9. Enable the Remote Attestation agent.

```
SLX (config-remote-attestation) # agent-enable
SLX (config-remote-attestation) #
```

10. Exit out of the Remote Attestation configuration mode.

```
SLX (config-remote-attestation) # exit
SLX (config)#
```

11. Exit out of the Global Configuration mode.

```
SLX (config)# exit
SLX #
```



Network Time Protocol (NTP)

[Network Time Protocol overview](#) on page 168

[Configuring NTP](#) on page 172

[Authenticating an NTP server](#) on page 173

[Displaying the active NTP server](#) on page 174

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
```

**Note**

After upgrading to SLX-OS 20.1.x, a downgrade to a previous version of the SLX-OS which does not provide support for an NTP source interface will remove the NTP server configuration. This will require you to reconfigure the NTP server command after the downgrade.

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 175

[Configuring SNMPv2](#) on page 181

[Configuring SNMPv3](#) on page 182

[Configuring an SNMP server context to a VRF](#) on page 183

[Offline SNMP ifIndex generation tool](#) on page 184

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 receives 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 Networks 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 Networks 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
```



Important

SNMP SET operation is not supported.

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.

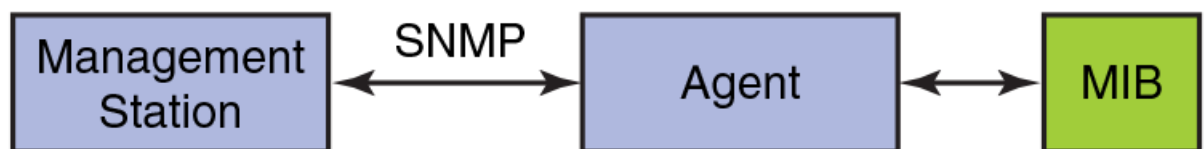


Figure 9: SNMP structure

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.

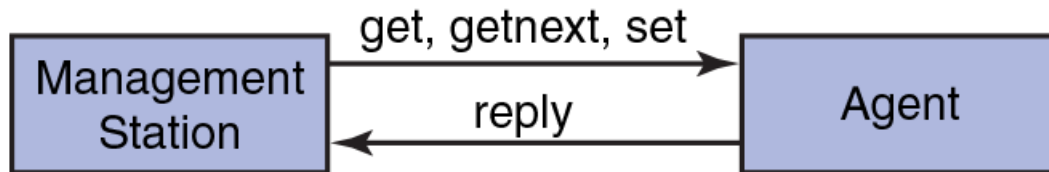


Figure 10: SNMP query

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 11: 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.

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.

SNMP Engine ID

On the Extreme device, the SNMP Engine ID can be configured as either a 12 byte or 13 byte ID. It is also now possible to view the default SNMP Engine ID through the CLI

show run command. Manually configured SNMP Engine ID are also reflected within the running configuration.

SNMP agent Engine ID can be 12 or 13 bytes in hexadecimal format. Each byte must be separated by colons. When using a 12 byte engine ID, each byte must be entered as 2 char per octet. For example, the value 7 must be entered as 07. However, for the 13 byte engine ID, each byte can be entered as 1 or 2 char per octet. For example, the value 7 can be entered as 7 or as 07.

Note the following when working with SNMP Engine ID:

- The Default SNMP Engine ID is 13 bytes long.
- You can view the Default SNMP Engine ID using the **show running config** command. You can also fetch this SNMP Engine ID through Netconf query and **snmpget** commands.
- Manually configured SNMP Engine ID is reflected within the running configuration and immediately accessible with the netconf query. The manually configured SNMP Engine ID will only be available through **snmpget** command and in the traps only after a successful reboot.
- When you un-configure the Default SNMP Engine ID, the running configuration will still retain the Default SNMP Engine ID. However, when you un-configure a Manually configured SNMP Engine ID, the running configuration will show the Default SNMP Engine ID. This change will happen only after a successful reboot.
- When a device is reloaded with a default configuration, the running configuration will display the Default SNMP Engine ID.
- When a manually configured SNMP Engine ID is reset, a syslog is generated. For example,

```
2020/12/18-03:00:56, [SNMP-1005], 77,, INFO, SLX, SNMP configuration attribute,
LocalEngineId, has changed from
[a1:b1:c1:d1:e1:a1:b1:c1:d2:a1:a1:b1] to [80:0:6:34:b2:4:0:0:10:aa:9a:b7:96].
```

- A syslog is not generated when the Default SNMP Engine ID is reset.
- When a 12 byte SNMP Engine ID is configured in version 20.2.3 and the device is then downgraded to a lower firmware release, this SNMP Engine ID is retained in the lower release post downgrade.
- When a 13 byte SNMP Engine ID is configured in version 20.2.3 and the *full install* downgrade is performed, this 13 byte SNMP Engine ID will not be available after downgrade. However, you can view this 13 byte SNMP Engine ID immediately after a *coldboot* downgrade and will be lost on subsequent file replays, config rollback, or copy of startup configuration.
- When the Default SNMP Engine ID is reset using the **no snmp-server engineID local** command, a warning message is displayed. This warning message is, however, not displayed when it is reset using the REST/Netconf query.

```
SLX(config)# no snmp-server engineID local
80:0:6:34:b2:4:0:0:10:aa:9a:b7:96
```

```
%Warning: SNMP engine id is currently default. Removing default engine id would again
set it to default value%.
```

A reboot is necessary for the configured engine ID to become active.

Use the **no** form of the command to remove the configured engine ID from database.

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
privatel23 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.

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   : 00011000110000001100000000000001
```

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   : 000011000110000100000000011000001
```

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   : 00101000000000000000000000000001
```

VE interface

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i ve -vi 10 -d all
Decimal : 1207959562
Hex      : 4800000a
Binary   : 010010000000000000000000000001010
```

Tunnel interfaces

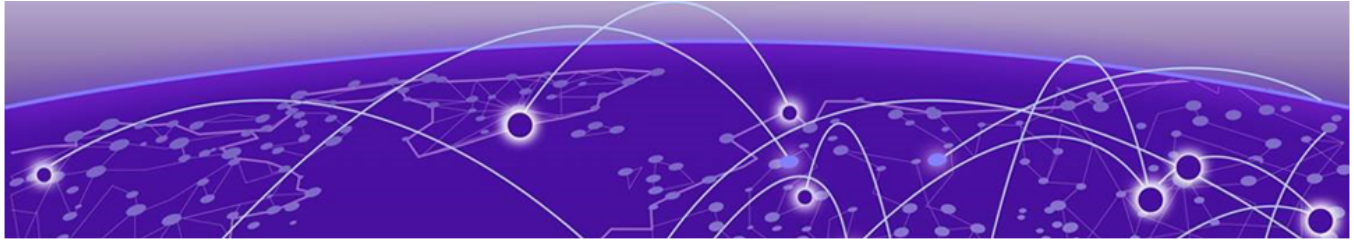
```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i tunnel -tt mpls -ti 2 -d all
Decimal : 2092957698
Hex      : 7cc00002
Binary   : 011111001100000000000000000000010
```

Loopback interface

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i lb -lbi 20 -d all
Decimal : 1476395028
Hex      : 58000014
Binary   : 0101100000000000000000000000010100
```

Management interface

```
[admin@SLX]# /fabos/cliexec/ifindex_gen.py -i mgmt -mi 2 -d all
Decimal : 805306370
Hex      : 30000002
Binary   : 001100000000000000000000000000010
```



LLDP

[LLDP overview](#) on page 187

[Configuring and managing LLDP](#) on page 190

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 Networks 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 Networks LLDP implementation adds a proprietary Extreme Networks 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 Networks 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 Networks devices.

Understanding the default LLDP

The following table lists the default LLDP configuration. Consider this when making changes to the defaults.

Table 25: 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 Networks does not recommend advertising dot1.tlv and dot3.tlv LLDPs if your network contains CNAs from non-Extreme Networks 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.

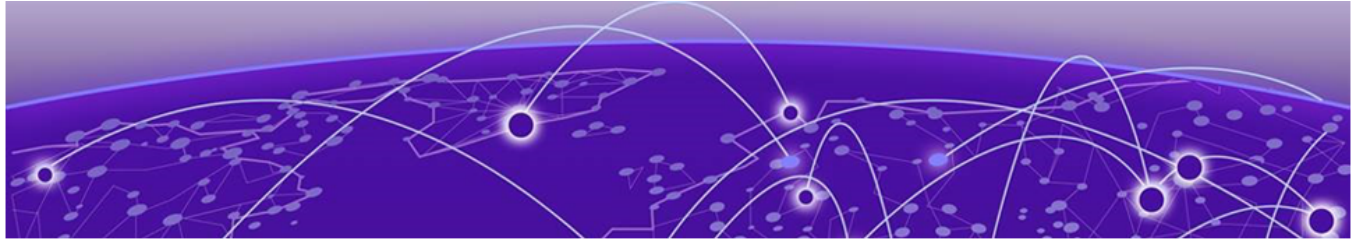
2. Clear the LLDP statistics on an interface.

```
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 197

[Root account and password recovery](#) on page 197

[Force Password Change At First Login](#) on page 200

[Force Password Age Out](#) on page 200

[Configure an account to disable automatically](#) on page 201

[Configure an account with inactivity warning](#) on page 202

[Changing default password for the system default accounts](#) on page 202

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.
 - [Recover the root login account](#) on page 198



Note

- For SLX 9540 running SLX-OS 20.1.x and lower, use the steps described in [Recover root login when shell prompt is available](#). The outputs and steps listed in the topic [Recover root login when ONIE is available](#) are not applicable in this case.
 - For SLX 9540 running version SLX-OS 20.2.x and higher and all other SLX platforms, such as Extreme 8720, use the steps described in [Recover root login when ONIE is available](#) for recovering the root password.
- When the root account is enabled but the root password is not available, perform the relevant task:
 - [Recover the root password when shell prompt is available](#) on page 199
 - [Recover the root password when ONIE is available](#) on page 200



Note

The default password for the root account on the VM is `fibranne`.

Recover the root login account

If the root account is disabled and SLX-OS CLI is not available, recover the root login account.



Note

These instructions are for all devices except SLX 9540 running SLX-OS 20.1.x and below versions.

For instructions to restore root login/password on SLX 9540 running SLX-OS 20.1.x and below, see [Recover the root password when shell prompt is available](#) on page 199.

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 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.

Recover the root password when shell prompt is available

If you forgot the password for the VM root account and when *shell prompt* is available, you can recover the default password.



Note

To perform the recovery process, you will need access to the shell prompt. The *shell prompt* is only available in SLX 9540 when the platform is running version SLX-OS 20.1.x and below.

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?
```

2. Choose option 3 to access the uboot prompt.

```
Option? 3
=>
```

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.

4. 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
=>
```

5. 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.

Recover the root password when ONIE is available

If you forgot the password for the VM root account and ONIE mode is available, you can recover the default password.



Note

ONIE is available for all SLX devices, except for SLX 9540 running SLX-OS version 20.1.x and below.

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 RootPasswd
```

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 `fibranne` password.

Force Password Change At First Login

To increase security, it is recommended that the default, factory password be changed immediately. This section describes how to force users, including admin users, to change their login after first successful login. This is not applicable to the `root` user.

Perform the following steps to force change of password on first login for all user accounts (admin as well as user).

1. Open a session to access the device.
2. Log in as admin.
3. Access global configuration mode.

```
device# configure terminal
```

4. Configure the setting to force users to change password.

```
device(config)# password-attributes force-default-password-change
```

When the user logs in for the first time, they will be forced to change their password.

Force Password Age Out

To increase security, it is recommended that password for all accounts be changed frequently. This section describes how to force users, including admin users, to

change their passwords on expiry of a pre-configured time interval. This is a global configuration.

Perform the following steps to force change of password on expiry of a pre-configured time interval.

1. Open a session to access the device.
2. Log in as admin.
3. Access global configuration mode.

```
device# configure terminal
```

4. Configure the setting to enforce changing of password after expiry of a set time period in days. This time duration is called *Age Out* duration.

```
device(config)# password-attributes max-password-age 90
```

This example configures a password's maximum age as 90 days. Each user is forced to change the password every 90 days. This is a global configuration and is applicable to all users configured on the system.

Configure an account to disable automatically

When creating or editing an account, you can specify when the account automatically disables after it is not used (active) for a configured period of time.

There might be instances when you would like to automatically disable an account when the account is inactive for some set period of time. Inactivity means that the account has not been used, in the recent past, to access this device. Use the **acct-inactivity-expiry-period** parameter to configure the number of days after which the account is automatically disabled (expires).



Note

The *root* and *admin* accounts cannot be disabled.

SNMP traps are generated for this event. For more information, see the *Extreme Message Reference*, 20.3.3. The traps generated are SEC-3138 and SEC-3139.

1. In privileged EXEC mode, enter the **configure terminal** command.

```
SLX # configure terminal
```

2. Enter the **username** command with the **acct-inactivity-expiry-period** parameter along with the number of days of inactivity, after which the account will automatically be disabled.

```
SLX (config)# username aming role user password Tijdlspw acct-inactivity-expiry-period 30
```

The account *aming* is now configured to automatically expire after 30 continuous days of inactivity. This is calculated from the day the account was created or from the last login. Expiry RASLOG is generated when time crosses the acct inactivity expiry period.

Configure an account with inactivity warning

When defining or editing an account that automatically expires, you can specify a duration after which a warning is generated about the inactivity of the account.

By default, users are not warned about the inactivity of their account. Use the **acct-inactivity-warning-period** parameter to configure the number of days after which a warning is generated about the account being inactive. For example, when set to 20 days, a warning will be generated when a specific user account is inactive for 20 days.



Note

Without configuring **expiry** period, **warning** period cannot be configured.

1. In the privileged EXEC mode, enter the **configure terminal** command.

```
SLX # configure terminal
```

2. Enter the **username** command with the **acct-inactivity-warning-period** command with the number of days.

```
SLX (config)# username aming role user password Tijdlspw acct-inactivity-warning-period 20
```

The account *aming* is now configured to generate a warning after 20 continuous days of the account being inactive. Warning RASLOG is generated when time crosses the account inactivity warning period.

Changing default password for the system default accounts

The system default accounts on the device have default passwords preconfigured. For security reasons, Extreme recommends changing these passwords from their defaults immediately after bringing up the device.

The required parameters for changing the password for the system default accounts and the user defined accounts are *name* and *password*. All other parameters are optional.

To enforce that the user changes the default password at first login, use the command **password-attributes force-default-password-change**.

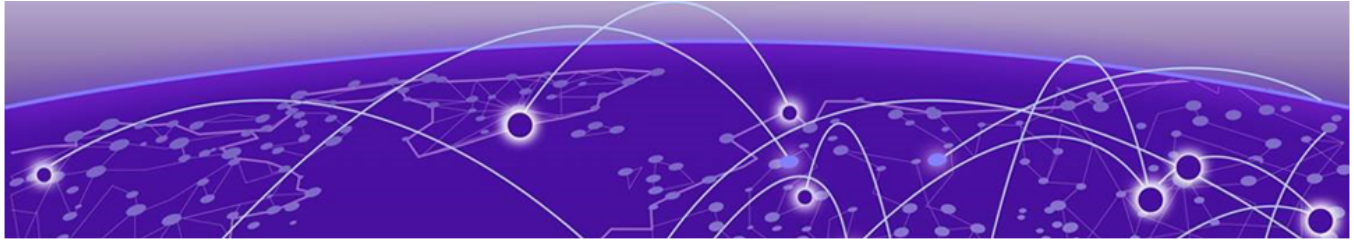
1. In the Privileged EXEC Mode, enter the **configure terminal** command.

```
SLX # configure terminal
SLX (config)#
```

2. Enter the **username** with the specified parameters. Use the **encryption-level** with 0 to indicate that this is a plain text password.

```
SLX (config)# username admin role admin password j*L4$hTS4 encryption-level 0
```

The default password for the *admin* account is now modified.



Python Event-Management and Scripting

[Python under Extreme operating systems](#) on page 203

[Python scripts](#) on page 207

[Python event-management](#) on page 214

[Troubleshooting event-management](#) on page 217

[Event-management show commands](#) on page 217

Python under Extreme operating systems

The Python interpreter installed with supported Extreme Networks 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 Networks 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>.

Launching Python interface from SLX-OS

You can launch the Python interface from within the SLX-OS CLI.

The Python interface provides additional features such as running Python commands and scripts to enable better and easier management of your SLX-OS device.

To launch the Python interface:

1. From the Privilege Execution Mode, use the **python** command to launch the python interface.

```
device# python
```

- A disclaimer is immediately displayed. The content of this disclaimer may change from time to time.
- This disclaimer is not displayed when using the -c option to run your Python scripts or commands.

```
device# python
Python 3.5.5 (default, Jan 19 2021, 08:38:37)
[GCC 7.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.

Disclaimer for Python shell usage!

The Python shell access on this SLX device is intended for diagnostics and
debugging purposes solely by the equipment vendor's trained engineers.
Improper use of the functionality made available through Python shell
access could cause significant harm and disruption to the network operation.

Your use of the functionality made available through Python shell access
is at your sole risk and you assume all liability resulting from such use.
The equipment vendor shall have no liability for any losses or damages
arising from or relating to Python shell access (and the functionality
enabled thereby) by anyone other than the equipment vendor's
authorized engineers.

Proceeding with the usage of Python shell access on this device explicitly
indicates your agreement to the terms of this disclaimer.

>>>
```

The interactive Python interface is now ready to use.



Caution

A disclaimer is always displayed when the Python shell is launched except when running with the -c option. Executing Python commands using the CLI() function or using the -c option remains a violation even if the disclaimer is not displayed.

2. The interactive Python interface provides comprehensive online help accessible from within the interface. Use the **help()** python command to do so.

```
>>> help()
```

The following information is displayed.

```
>>> help()

Welcome to Python 3.5's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at http://docs.python.org/3.5/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".
```

```
To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics". Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".
```

```
help>
```

Use the **quit** command to exit the interactive help session and return you to the Python interface.

```
help> quit
>>>
```

3. To exit out of the interactive Python interface, use the **quit()** python command.

```
>>> quit()
device#
```

The interactive Python session is closed and you are returned to the SLX-OS Privilege Execution Mode prompt.

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 Networks 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 Networks operating system prompt, enter either:
 - `exit()`
 - `Ctrl-D`
3. To run a Extreme Networks 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 Networks 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 Networks operating system CLI commands in the Python shell rather than in the standard Extreme Networks 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 Networks 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 210.

Testing Python-script statements

While developing a Python script, you can test Extreme Networks 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 Networks device.



Important

The only supported USB device for this task is the Extreme Networks 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 Networks 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 Networks 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 214:

```
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.

4. 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.

5. 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
```


6. 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.

1. 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
```

2. 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 26: 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 218

[Configuration rollback details](#) on page 220

[Configuration rollback considerations and limitations](#) on page 221

[Configuring rollback](#) on page 223

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.

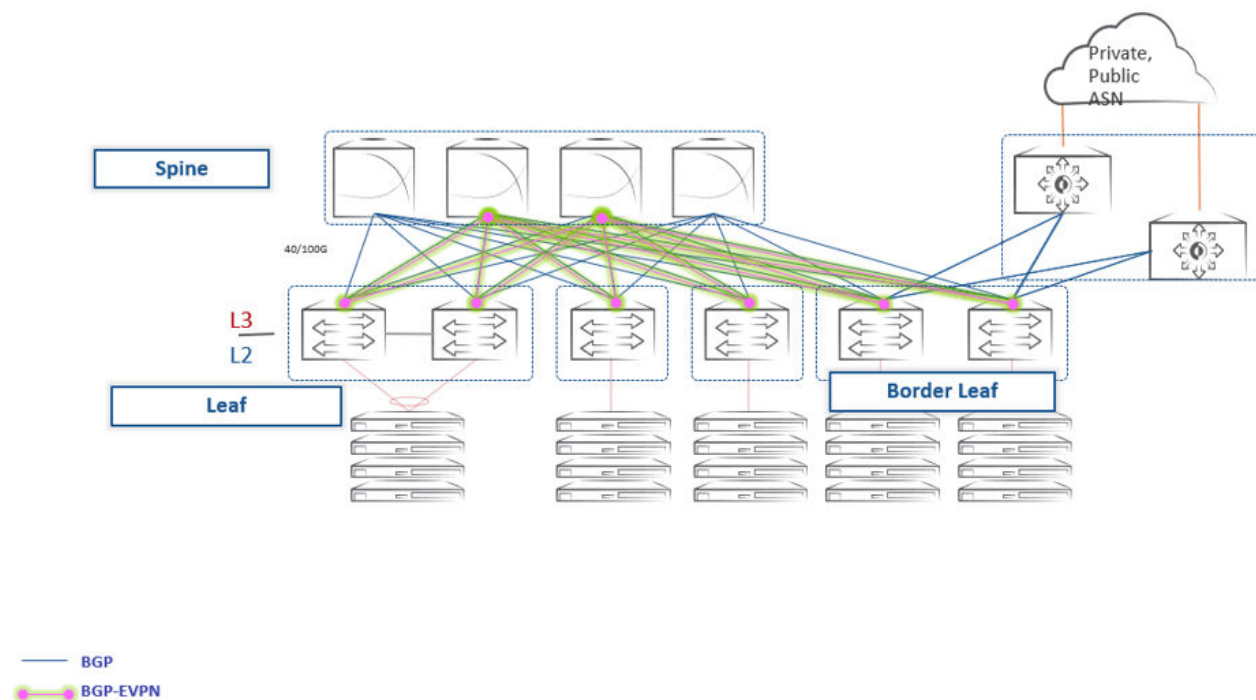


Figure 12: 3-tier IP Clos topology

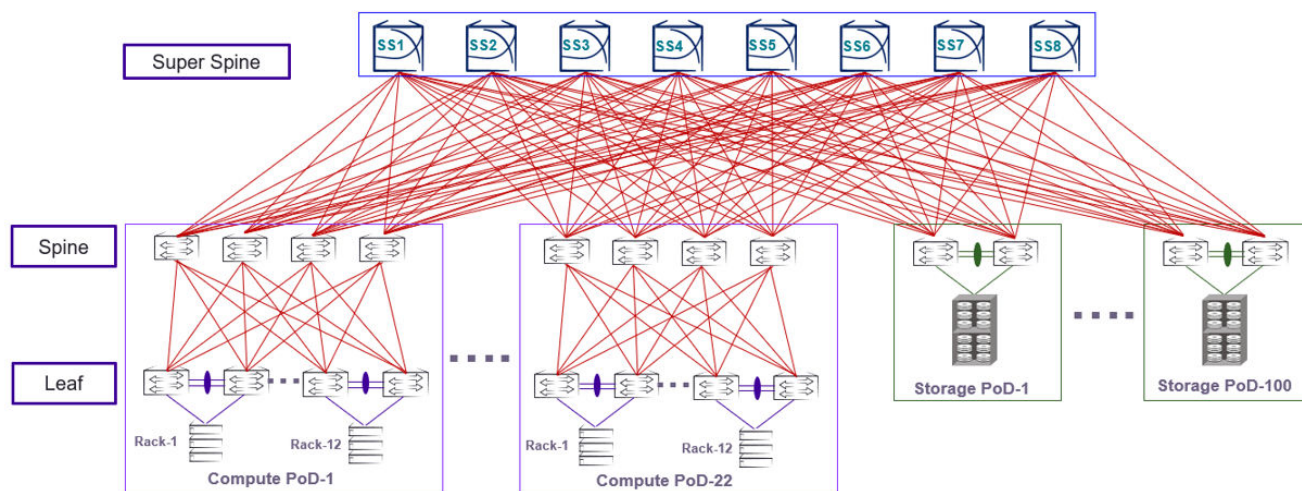


Figure 13: 5-tier IP Clos topology

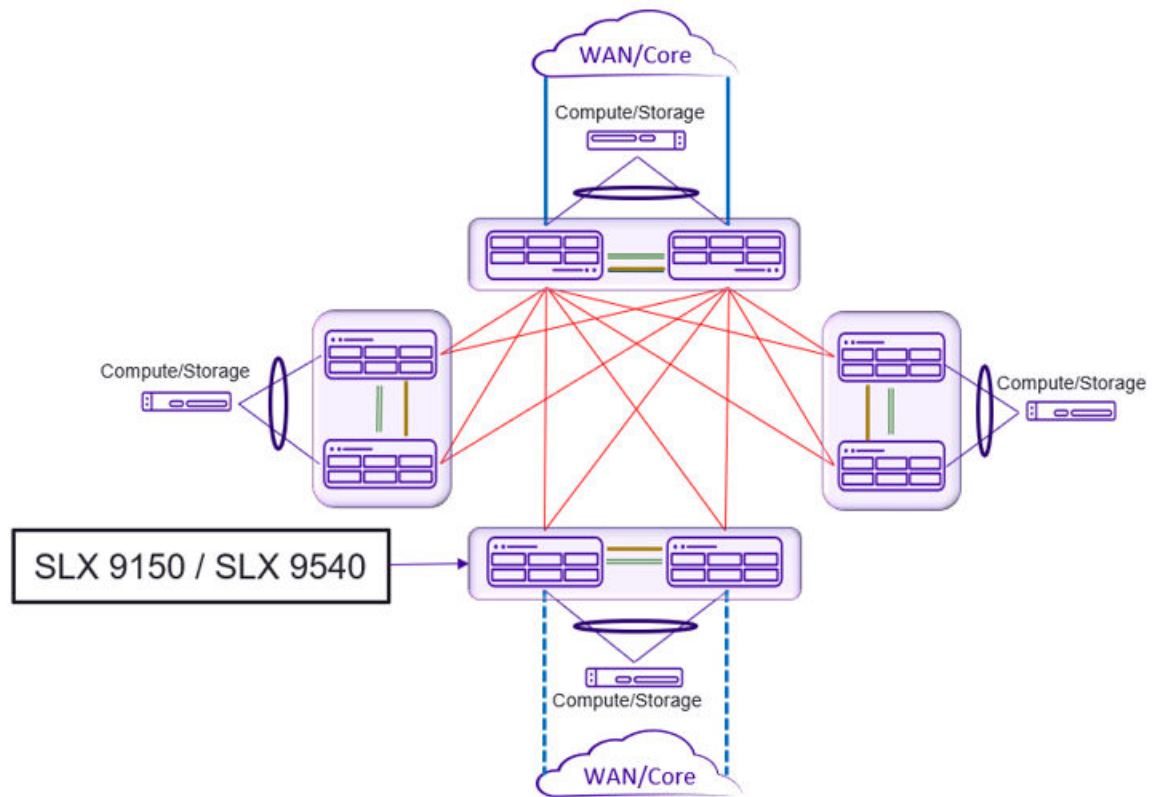


Figure 14: Non-Clos 4-rack topology

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_HHMMSS* 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 27: 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.

Table 27: RAS conditions and messages (continued)

Condition	Message
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 evl
  action python-script show_interface.py
Error: flash://show_interface.py script for event handler evl 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
```



BMC Configuration

- [Increase BMC Security](#) on page 228
- [Change BMC User Password](#) on page 229
- [Enable the BMC Management Interface](#) on page 231
- [Configure BMC Management Interface IP Address](#) on page 231
- [Managing BMC Management Interface](#) on page 233
- [Reset BMC Configuration to Factory Defaults](#) on page 234

Increase BMC Security

This topic discusses the steps to increase BMC's security including changing the password for the default account, and changing the default IP address for the BMC's network interface.

Intelligent Platform Management Interface

Intelligent Platform Management Interface (IPMI) is a set of specifications that defines how to manage and monitor a device independent of its Operating System (OS), underlying Hardware, and the BIOS installed on it. IPMI also defines a set of physical interfaces that enable system administrators to perform *out-of-band* management of IPMI capable devices, including such devices that have been powered off or that have network issues or are unresponsive. Without IPMI, a system administrator would need to be physically present near the device to resolve any issue.

IPMI is a message-based, hardware-level interface specification which exists and operates independently of the underlying operating system or the device's hardware. This enables IPMI to remotely manage a device even if the device does not have an installed OS. IPMI can also be used in scenarios where the device is powered down or even if there is a system or OS failure.

The target device can be powered down, however, for IPMI to work, it must at least be connected to an underlying local area network (LAN) and must be connected to a working power source.

IPMI can also be used to continuously monitor various statuses and statistics, such as temperature, fan speed, voltages, power supply status and physical access to the device.

Baseboard Management Controller

Baseboard Management Controller (BMC) is a dedicated microcontroller embedded on a device's motherboard and has its own dedicated firmware, RAM, and network port. Sensors on the motherboard transmit data to the BMC which in turn transmits this data to dedicated centralized monitoring servers. When connected to a LAN, the network port on the BMC enables *out-of-band* control and monitoring of the underlying hardware.

BMC enables IPMI on a device.

Securing BMC

BMC ships with a well known default User ID, password, and network configuration configured during firmware install at the factory. This provides an security vulnerability that can be exploited to gain access to the device.

Securing BMC involves changing the default User ID's password and changing the default network configuration. SLX-OS provides commands that interacts with the underline BMC firmware to harden the security of your device's BMC.

Change BMC User Password

BMC firmware ships with the well known default User ID of 2 with the password set to *qct.admin*. This information being well know, is a huge vulnerability and the best practice is to change the default password as early as possible.

To change the password for a specific BMC user:

1. Navigate into the BMC configuration mode.

```
SLX # configure terminal
SLX (config)# bmc
SLX (config-bmc)#
```

Configuration of each User ID must be done separately from within its configuration mode.



Note

Only the user with the User ID 2 can be configured. Though the command allows configuration of other User IDs, those changes are not applied on the BMC.

You are now within the BMC configuration mode.

2. From the BMC mode prompt, execute the **user-id** command to navigate to its configuration mode.

```
SLX (config-bmc)# user-id 2
SLX (config-bmc-user-2)#
```

Each User ID has its own configuration mode. Configurations made within this mode is only applicable to that specific User ID.

You are now within the configuration mode for the specific User ID.

3. Execute the **password** command along with the password to be set for the user.

```
SLX (config-bmc-user-2)# password testing123
SLX (config-bmc-user-2)#
```

Passwords must meet certain pre-defined criteria before it is configured for the User ID.

The password for the User ID is modified.

4. (Optional) To verify the password has changed successfully, use the **show bmc password-status** command.

```
SLX (config-bmc-user-2)# do show bmc password-status 2
Status: Set
SLX (config-bmc-user-2)#
```

This example shows the configuration of a new password for the User ID 2.

```
SLX # configure terminal
SLX (config)# bmc
SLX (config-bmc)# user-id 2
SLX (config-bmc-user-2)# password testing123
SLX (config-bmc-user-2)# exit
SLX (config-bmc)# exit
SLX (config)# exit
SLX #
```

The following example shows the output when you try to assign a password with length shorter than the minimum password length

```
SLX # configure terminal
SLX (config)# bmc
SLX (config-bmc)# user-id 2
SLX (config-bmc-user-2)# password test123
% Error: BMC password length should be 8-20 characters
SLX (config-bmc-user-2)#
```

You can also use the **do show running-config bmc user-id** command to view the encrypted form of the password.

```
SLX (config-bmc-user-2)# do show running-config bmc user-id
bmc
  user-id 2
    password $9$VaXhc9WCy+1IwRU2ZaS2vQ==
  !
SLX (config-bmc-user-2)#
```

Enable the BMC Management Interface

By default, the BMC Management Interface is disabled in SLX-OS devices. It must be manually enabled and configured.

To enable the BMC Management Interface:

1. Navigate to the BMC Management Interface context.

```
SLX# configure terminal
SLX (config)# bmc
SLX (config-bmc)#
```

**Note**

Only the BMC Management Interface with interface ID of 0 (zero) can be configured.

You are now within the BMC configuration mode.

2. From within the BMC configuration mode, navigate to the BMC Management Interface configuration mode.

```
SLX (config-bmc)# interface management 0
SLX (config-bmc-mgmt-0)#
```

You are now within the BMC Management Interface configuration mode.

3. Enable the interface.

```
SLX (config-bmc-mgmt-0)# no shutdown
SLX (config-bmc-mgmt-0)#
```

The BMC Management Interface is now enabled and ready to be configured for out-of-band access.

4. (Optional) Verify if the BMC Management Interface is enabled.

```
SLX (config-bmc-mgmt-0)# do show bmc interface management 0 status
Status: Enabled

SLX (config-bmc-mgmt-0)#
```

Configure BMC Management Interface IP Address

BMC firmware ships with the well known default configuration of DHCP for the BMC Management Interface. Once SLX-OS firmware is downloaded and installed, SLX-OS configures it predefined static IP address to the BMC Management Interface.

Keep the IPv4 address, the Netmask, and the Gateway IPv4 address that is required to be configured, ready.

**Note**

SLX-OS configures the following static IP 192.168.11.1/24 and default gateway 0.0.0.0 by default.

To configure the BMC Management Interface:

1. Navigate into the BMC Management Interface context.

```
SLX # configure terminal
SLX (config)# bmc
SLX (config-bmc)#
```



Note

Only the BMC Management Interface with interface ID of 0 can be configured.

You are now within the BMC configuration mode.

2. From within the BMC configuration mode, navigate into the BMC Management Interface configuration mode.

```
SLX (config-bmc)# interface management 0
SLX (config-bmc-mgmt-0)#
```

You are now within the BMC Management Interface configuration mode.

3. Configure the IPv4 address for the BMC Management Interface.

- To configure the BMC Management Interface to receive the IPv4 address from a remote DHCP server, use the following command.

```
SLX (config-bmc-mgmt-0)# ip dhcp
SLX (config-bmc-mgmt-0)#
```

- To configure the BMC Management Interface's IPv4 address manually, execute the following command:

```
SLX (config-bmc-mgmt-0)# ip address 10.9.9.23/24 gw 10.9.9.2
SLX (config-bmc-mgmt-0)#
```

The IPv4 address for the BMC Management Interface is either automatically assigned or manually configured.

4. (Optional) Verify by issuing the **do show bmc interface management 0 ip** command.

```
SLX (config-bmc)# do show bmc interface management 0 ip
IP Address Source : Static Address
IP Address       : 10.9.9.23
Subnet Mask      : 255.255.255.0
MAC Address      : 00:04:96:b8:41:b8
Default Gateway IP : 10.9.9.2
SLX (config-bmc-mgmt-0)#
```

This example shows the configuration of DHCP for the BMC Management Interface.

```
SLX # configure terminal
SLX (config)# bmc
SLX (config-bmc)# interface management 0
SLX (config-bmc-mgmt-0)# ip DHCP
SLX (config-bmc-mgmt-0)# exit
SLX (config-bmc)# do show bmc interface management 0 ip
IP Address Source : DHCP Address
IP Address       : 192.168.0.120
Subnet Mask      : 255.255.255.0
MAC Address      : 00:04:96:b8:41:b8
Default Gateway IP : 0.0.0.0
SLX (config-bmc-mgmt-0)#
```


This example shows the default configuration of a static IPv4 address. This IP address is assigned to the interface automatically and will be retained unless changed.

```
SLX # configure terminal
SLX (config)# bmc
SLX (config-bmc)# interface management 0
SLX (config-bmc-mgmt-0)# no ip DHCP
SLX (config-bmc-mgmt-0)# exit
SLX (config-bmc)# do show bmc interface management 0 ip
IP Address Source : Static Address
IP Address       : 192.168.11.1
Subnet Mask      : 255.255.255.0
MAC Address      : 00:04:96:b8:41:b8
Default Gateway IP : 0.0.0.0
SLX (config-bmc-mgmt-0)#
```

Managing BMC Management Interface

BMC firmware ships with the well known default configuration of DHCP for the BMC Management Interface. Once SLX-OS firmware is downloaded and installed, SLX-OS configures it predefined static IP address to the BMC Management Interface. SLX-OS configures the following static IP 192.168.11.1/24 and default gateway 0.0.0.0 by default.

To enable or disable out-of-band access to BMC Management Interface:

1. Navigate into the BMC Management Interface context.

```
SLX # configure terminal
SLX (config)# bmc
SLX (config-bmc)#
```

Configuration of each BMC Management Interface must be done separately from within its configuration mode.



Note

Only the BMC Management Interface with interface ID of 0 can be configured.

You are now within the BMC configuration mode.

2. From within the BMC configuration mode, navigate into the BMC Management Interface configuration mode.

```
SLX (config-bmc)# interface management 0
SLX (config-bmc-mgmt-0)#
```

You are now within the BMC Management Interface configuration mode.

3. To set the out-of-band access to the BMC Management Interface:

- If you need to shutdown the interface to prevent out-of-band access, use the **shutdown** command.

```
SLX (config-bmc-mgmt-0) # shutdown
SLX (config-bmc-mgmt-0) #
```

- If the interface is in the *shutdown* state, you can enable it by executing the **no shutdown** command.

```
SLX (config-bmc-mgmt-0) # no shutdown
SLX (config-bmc-mgmt-0) #
```

Verify the status of the interface using the **do show running-config** command.

This example shows how to shutdown the BMC Management Interface to prevent out-of-band access.

```
SLX # configure terminal
SLX (config) # bmc
SLX (config-bmc) # interface management 0
SLX (config-bmc-mgmt-0) # shutdown
SLX (config-bmc-mgmt-0) # exit
SLX (config-bmc) # exit
SLX (config) # do show bmc interface management 0 status
Status: Disabled
```

This example shows how to enable a shutdown BMC Management Interface to enable access to it from out-of-band ports.

```
SLX # configure terminal
SLX (config) # bmc
SLX (config-bmc) # interface management 0
SLX (config-bmc-mgmt-0) # no shutdown
SLX (config-bmc-mgmt-0) # exit
SLX (config-bmc) # exit
SLX (config) # do show bmc interface management 0 status
Status: Enabled
```

Reset BMC Configuration to Factory Defaults

In case of a security breach, it might be required that the configuration applied to a BMC's be reset to the factory defaults. Use the `bmc factory reset` command to do so.

To reset the device's BMC configuration back to its factory defaults:

1. From the Executable Mode, execute the command `bmc factory reset`.

```
SLX# bmc factory reset
SLXOS needs to be restarted for normal operations, after bmc factory reset.
Do you want to continue? (y/n):
```

2. Enter `y` to proceed with restarting the device and resetting the BMC configuration to factory defaults.

```
SLX# bmc factory reset
SLXOS needs to be restarted for normal operations, after bmc factory reset.
Do you want to continue? (y/n): y
SLX#
```

3. Use the **reload** command to force the device to reboot.

```
SLX# reload
```

The device starts to reboot and post reboot, the BMC configuration is reset to its factory defaults.

The following example shows the reset to BMC's factory default settings.

```
SLX# bmc factory reset
SLXOS needs to be restarted for normal operations, after bmc factory reset.
Do you want to continue? (y/n): y
SLX# reload
...
```