Extreme SLX-OS
Security Configuration Guide, 20.3.1

Supporting ExtremeRouting and ExtremeSwitching SLX 9740, SLX 9640, SLX 9540, SLX 9250 and SLX 9150

Table of Contents

Preface

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

Text Conventions

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

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

Table 1: Notes and warnings

Icon	Notice type	Alerts you to...
💡	Tip	Helpful tips and notices for using the product
📝	Note	Useful information or instructions
➡	Important	Important features or instructions
⚠	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 *enter* and *type*	When you see the word *enter* 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 *type*.
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**
Words in italicized type	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.
italic 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, `member[member...]`.
\	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 white papers, data sheets, and case studies

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

Getting Help

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

Extreme Portal

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

The Hub

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

Call GTAC

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

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

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

Subscribe to 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.

Providing Feedback

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

- Content errors, or confusing or conflicting information.

- Improvements that would help you find relevant information in the document.
- Broken links or usability issues.

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

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

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

About this document

What's New in this Document

The following table includes descriptions of new information added to this document for the SLX-OS 20.3.1 software release.

Table 4: Summary of changes

Feature	Description	Described in
ACL overview	Modified the section *ACL overview* and removed references to *uda* and *uda offsets* as they are not supported in SLX-OS 20.3.1.	ACL overview on page 30
Configuration guidelines for bACLs	Modified the section *Configuration guidelines for bACLs* and removed references to *uda* and *uda offsets* as they are not supported in SLX-OS 20.3.1.	Configuration guidelines for bACLs on page 72

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 20.3.1 supports the following hardware platforms.
- Devices based on the Broadcom XGS® chipset family:
  - ExtremeSwitching SLX 9250
  - ExtremeSwitching SLX 9150
- Devices based on the Broadcom DNX® chipset family:
  - ExtremeRouting SLX 9740
  - ExtremeRouting SLX 9640

- ○ ExtremeSwitching SLX 9540

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

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

User Accounts and Passwords

User account overview

A user account specifies that user's level of access to the device CLI.

The software uses role-based access control (RBAC) as the authorization mechanism. A *role* is a container for rules, which specify which commands can be executed and with which permissions. When you create a user account you need to specify a role for that account. In general, *user* (as opposed to *user-level*) refers to any account—to which any role can be assigned—user, admin, or a non-default role.

Default accounts and roles

The software ships with two default accounts—admin and user—and two corresponding default roles:

- `admin`—Accounts with admin permissions can execute all commands supported on the device. (For the initial admin login, refer to the relevant *Hardware Installation Guide*.)

- `user`—Accounts with user-level permissions can execute all `show` commands supported on the device. User-level accounts can also execute the following operational commands: `cfm`, `execute-script`, `exit`, `mtrace`, `no`, `ping`, `rasman`, `ssh`, `sysmon`, `telnet`, `timestamp`, `trace-l2`, and `traceroute`.

  Note
  For details on non-default roles (also known as *user-defined roles*), refer to User-defined roles on page 17.

Account guidelines and limitations

Be aware of the following guidelines and limitations:

- Extreme recommends that every user access the CLI through a unique account: After logging in as admin, create a unique account for yourself, specifying **role admin**.
- You cannot modify rules for the admin or the user default accounts.
- You cannot modify rules for the admin or the user default roles.
- By default, all account information is stored in the device-local user database.
- By default, user authentication and tracking of logins to the device is local.
- The maximum number of accounts—including the two default accounts—is 64. For more than 64 users, you can implement an authentication, authorization, and accounting (AAA) service. For details, refer to the External Server Authentication section.
- The maximum number of roles—including the two default roles—is 64. If needed, refer to Role and rule limits on page 17.

Basic account management

These topics enable you to create and manage basic admin and user accounts.

Creating an admin-role account

An admin-role account can execute all supported CLI commands.

The required parameters for creating an account are **name**, **role**, and **password**. In this example, the optional **desc** parameter is also utilized.

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

   ```
   device# configure terminal
   ```

2. Enter the **username** command, with the specified parameters.

   ```
   device(config)# username jsmith role admin password Tijd1spw desc "Has access to all
   commands"
   ```

Creating a user-role account

A user-role account can execute **show** and other basic CLI commands.

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

   ```
   device# configure terminal
   ```

2. Enter the **username** command, with the specified parameters.

   ```
   device(config)# username jdoe role user password iKt1Sas*p
   ```

Modifying an account

Use this topic to modify a user account.

The only required parameter for modifying an account is **username** *username*. In this example, the role is changed to admin.

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

   ```
   device# configure terminal
   ```

2. Enter the **username** command, with the needed parameters.

   ```
   device(config)# username jdoe role admin
   ```

Disabling an account

Use this topic to disable a user account.

> **Note**
> If you disable an account, all active sessions for that user are immediately terminated.

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

   ```
   device# configure terminal
   ```

2. Enter the **username** command, with the **enable false** parameters.

   ```
   device(config)# username testUser enable false
   ```

Unlocking an account

Use this topic to unlock a user account.

A user account is automatically locked by the system when the configured threshold for repeated failed login attempts has been reached. The account lockout threshold is a configurable parameter. Refer to Account lockout policy on page 26 for more information.

> **Note**
> The **username** and **no username** commands are global configuration commands, but the **unlock username** command is a privileged EXEC command.

1. In privileged EXEC mode, enter the **show users** command to display currently active sessions and locked out users.

   ```
   device# show users
   **USER SESSIONS**
   Username    Role    Host IP      Device    Time Logged In
   jsmith      user    192.0.2.0    Cli       2016-04-30 01:59:35
   jdoe        admin   192.0.2.1    Cli       2016-05-30 01:57:41

   **LOCKED USERS**
   testUser
   ```

2. For each account that you want to unlock, enter the **unlock username** command.

   ```
   device#  unlock username testUser
   Result: Unlocking the user account is successful
   ```

3. Enter the **show users** command to verify that the account is unlocked.

```
device# show users
**USER SESSIONS**
Username    Role    Host Ip        Method    Time Logged In         TTY
jsmith      user    192.0.2.0      cli       2016-04-30 01:59:35    pts/2
jdoe        admin   192.0.2.1      cli       2016-05-30 01:57:41    tty80

**LOCKED USERS**
Username
no locked users
```

Deleting an account

Use this topic to delete a user account.

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

```
device# configure terminal
```

2. Enter the **no username** command.

```
device(config)# no username testUser
```

When an account is deleted, all active login sessions for that user are terminated

User-defined roles

In addition to the default roles—admin and user—the software supports the creation of user-defined roles.

User-defined-role overview

User-defined roles enable you to fine-tune CLI access.

A user-defined role starts from a basic set of privileges which are then refined by adding rules. You assign a name to the role and then associate the role to one or more user accounts.

The following tools are available for managing user-defined roles:

- The **role** command defines new roles and deletes user-defined roles.
- The **rule** command allows you to specify access rules for specific operations and assign these rules to a given role.
- The **username** command associates a given user-defined role with a specific user account.

Role and rule limits

At any given time, an account is associated with one role. A role is associated with one or more rules. A rule is associated with only one role.

This relationship among accounts, roles, and rules is illustrated by the following entity-relationship diagram:
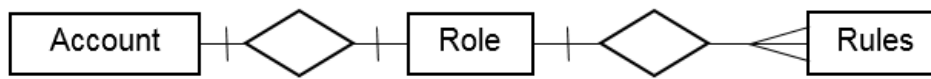
Figure 1: Accounts, roles, and rules

The number of supported accounts, roles, and rules is as follows:

- The maximum number of accounts is 64, including the default admin and user accounts. For more than 64 users, you can implement an authentication, authorization, and accounting (AAA) service.
- The maximum number of roles is 64, including the default admin and user roles.
- The maximum number of rules is 512, which you can allocate among your roles as you see fit.

Creating or modifying a role

Use this topic to create a role or to modify its Description.

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

   ```
   device# configure terminal
   ```

2. Enter the **role** command, specifying the role name and (optionally) a description.

   ```
   device(config)# role name NetworkAdmin desc "Manages security CLIs"
   ```

Deleting a role

Use this topic to delete a role.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

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

2. Enter the **no role** command with the specified parameters.

   ```
   device(config)# no role name NetworkAdmin
   ```

Command-access rules

Command authorization is defined in terms of rules that you associate with a user-defined role.

Rules define and restrict a role to access modes (*read-only* or *read-write* access), and beyond that can define permit or reject on specified command groups or individual commands. You can associate multiple rules with a given user-defined role, but you can associate only one role with any given user account.

The following rule parameters are mandatory:

- **index**—a unique index number
- **role**—the unique role with which you are associating the rule
- **command**—the command to which the rule applies

The following rule parameters are optional:

- **operation**—specifies the type of operation permitted (**read-only** or **read-write**). The default is **read-write**.

- **action**—specifies whether the user is accepted or rejected while attempting to execute the specified command. The default value is **accept**.

The following example creates and assigns four rules to a role named "NetworkAdmin".

```
device(config)# rule 70 action accept operation read-write role NetworkAdmin command
configure
device(config)# rule 71 action accept operation read-write role NetworkAdmin command copy
running-config
device(config)# rule 72 action accept operation read-write role NetworkAdmin command
interface management
device(config)# rule 73 action accept operation read-write role NetworkAdmin command
clear logging
```

> **Note**
> Rules cannot be added for commands that are not at the top level of the command hierarchy. For a list of eligible commands, type **?** after the **command** keyword.

Rules for configuration commands

The following rules govern configuration commands:

- If a role has a rule with a **read-write** operation and the **accept** action for a configuration command, the user associated with this role can execute the command and read the configuration data.

- If a role has a rule with a **read-only** operation and the **accept** action for a configuration command, the user associated with this role can only read the configuration data of the command.

- If a role has a rule with a **read-only** or **read-write** operation and the **reject** action for a configuration command, the user associated with this role cannot execute the command and can read the configuration data of the command.

Rules for operational commands

Rules can be created for the specified operational commands. By default, every role can display all the operational commands but cannot execute them. The **show** commands can be accessed by all the roles.

The following rules govern operational commands:

- If a role has a rule with a **read-write** operation and the **accept** action for an operational command, the user associated with this role can execute the command.

- If a role has a rule with a **read-only** operation and the **accept** action for an operational command, the user associated with this role can access but cannot execute the command.

- If a role has a rule with a **read-only** or **read-write** operation and the **reject** action for an operational command, the user associated with this role can neither access nor execute the command.

Rules for interface commands

Rules can be created for a specific instance of the interface-related configuration commands.

By default, every role has the permission to read the configuration data related to all the instances of the interfaces using the **show running-config interface** command.

The following rules govern interface commands:

- If a role has a rule with a **read-write** operation and the **accept** action for only a particular instance of the interface, users associated with this role can only modify the attributes of that instance.

- If a role has a rule with a **read-only** operation and the **accept** action for only a particular instance of the interface, users associated with this role can only read (using the **show running-config** command) the data related to that instance of the interface.

- If a role has a rule with a **read-write** operation and the **reject** action for only a particular instance of the interface, users associated with this role cannot execute and read the configuration data for that interface instance.

  In the following example, the rules are applicable only to a particular instance of the specified interface.

  ```
  device(config)# rule 60 action accept operation read-write role NetworkAdmin command
  interface ethernet 1/4
  device(config)# rule 65 action accept operation read-write role NetworkAdmin command
  interface port-channel 2
  device(config)# rule 68 role NetworkAdmin action reject command interface ethernet 3/4
  ```

- If a role has a rule with a **read-only** or **read-write** operation and the **reject** action for an interface or an instance of the interface, users associated with this role cannot perform **clear** and **show** operations related to those interfaces or interface instances. To perform **clear** and **show** operations, the user's role must have at least **read-only** and the **accept** permission. By default, every role has the **read-only** and **accept** permission for all interface instances.

  In the following example, NetworkAdmin users cannot perform **clear** and **show** operations related to all **ethernet** instances.

  ```
  device(config)# rule 30 action accept operation read-write role NetworkAdmin command
  interface ethernet
  ```

- If a role has a rule with **read-only** or **read-write** operation, and the **reject** action for an interface **ethernet** instances, users associated with this role cannot perform **clear** and **show** operations related to those instances. To perform **clear** and **show** operations related to **interface ethernet** instances, the role should have at least **read-only** and **accept** permission. By default, every role has the **read-only** or **accept** permission for all interface instances.

  In the following example, users associated with the NetworkAdmin role cannot perform some of the **clear** and **show** operations related to all **ethernet** instances.

  ```
  device(config)# rule 30 role NetworkAdmin action reject command interface ethernet
  ```

- The **dot1x** option under the **interface** instance submode can only be configured if the role has the **read-write** and **accept** permissions for both the **dot1x** command and **interface** instances.

In the following example, users associated with the CfgAdmin role can access and execute the **dot1x** command in **ethernet** instances.

```
device(config)# rule 16 action accept operation read-write role cfgadmin command
interface ethernet
device(config)# rule 17 action accept operation read-write role cfgadmin command dot1x
```

Configuring a placeholder rule

A rule created with the **no-operation** command does not enforce any authorization rules. Instead, you can use the **no-operation** instance as a placeholder for a valid command that is added later, as shown in the following example.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

```
device# configure terminal
```

2. Enter the **rule** command with the specified parameters and the **no-operation** keyword as a placeholder.

```
device(config)# rule 75 action reject operation read-write role NetworkAdmin command
no-operation
```

3. Enter the **rule** command with the specified command to replace the placeholder.

```
device(config)# rule 75 role NetworkAdmin command firmware
```

Rule-processing order

When a user executes a command, rules are searched in ascending order by index for a match and the action of the first matching rule is applied. If none of the rules match, command execution is blocked. If there are conflicting permissions for a role in different indices, the rule with lowest index number is applied.

As an exception, when a match is found for a rule with the **read-only** operation and the **accept** action, the system seeks to determine whether there are any rules with the **read-write** operation and the **accept** action. If such rules are found, the rule with the **read-write** permission is applied.

In the following example, two rules with action accept are present and rule 11 is applied.

```
device(config)# rule 9 operation read-only action accept role NetworkAdmin command aaa
device(config)# rule 11 operation read-write action accept role NetworkAdmin command aaa
```

Adding a rule

You add a rule to a role by entering the **rule** command with appropriate options. Any updates to the authorization rules will not apply to the active sessions of the users. The changes are applied only when users log out from the current session and log in to a new session.

The following example creates the rules that authorize the security administrator role to create and manage user accounts.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

```
device# configure terminal
```

2. Create a rule specifying read-write access to the global configuration mode.

```
device(config)# rule 150 action accept operation read-write role SecAdminUser command
config
```

3. Create a second rule specifying read-write access to the **username** command. Enter the **rule** command with the specified parameters.

```
device(config)# rule 155 action accept operation read-write role SecAdminUser command
username
```

4. "SecAdminUser" users can create or modify user accounts.

```
device# configure terminal
Entering configuration mode terminal
Current configuration users:
admin console (cli from 127.0.0.1) on since 2010-08-16 18:35:05 terminal mode

device(config)# username testuser role user password (<string>): ********
```

Changing a rule

The following example changes the previously created rule (index number 155) so that the **username** command is replaced by the **role** command.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

```
device# configure terminal
```

2. Enter the **rule** command, specifying an existing rule (index 155) and the role; and changing the **command** attribute to the **role** command.

```
device(config)# rule 155 role SecAdminUser command role
```

After changing rule 155, "SecAdminUser" users can execute the **role** command, but not the **username** command.

Deleting a rule

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

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

2. Enter the **no rule** command followed by the index number of the rule you wish to delete.

```
device(config)# no rule 155
```

After rule 155 is deleted, the SecAdminUser can no longer access the **role** command.

Advanced account management

These topics enable you to create non-default accounts and to configure advanced settings.

Creating a non-default account

The permissions for a non-default account are determined by the role assigned to it.

The required parameters for creating an account are **name**, **role**, and **password**. In this example, the optional **desc** parameter is also utilized.

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

```
device# configure terminal
```

2. Enter the **username** command, with the name, role, initial password, and optional parameters.

```
device(config)# username mlopez role NetworkAdmin password xL*84qt desc "Has access to
all network admin commands."
```

Creating an account with clock-restricted access

When defining or editing an account, you can specify permitted access hours.

By default, users can log in 24 hours a day. The **access-time** parameter enables you to limit access to defined hours, as per the system time defined for the operating system. For the current system time, enter **show clock**.

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

```
device# configure terminal
```

2. Enter the **username** command, with the **access-time** parameter.

```
device(config)# username aming role user password Tijd1spw access-time 0800 to 1800
```

Password policies

Password policies define and enforce a set of rules that make passwords more secure by subjecting all new passwords to global restrictions.

Password policies overview

You can configure password strength policy, password encryption policy, and account lockout policy.

The password policies described in this section apply to the device-local user database only.

Configured password policies (and all user account attributes and password state data) remain unchanged after an HA failover.

> **Note**
> For recovering the root password, refer to the *Extreme SLX-OS Management Configuration Guide*.

Password strength policy

The following table lists configurable password policy parameters.

Table 5: Password policy parameters

Parameter	Description
admin-lockout	Enables lockout for admin role accounts.
character-restriction lower	Specifies the minimum number of lowercase alphabetic characters that must occur in the password. The maximum value must be less than or equal to the minimum length value. The default value is zero, which means there is no restriction of lowercase characters.
character-restriction upper	Specifies the minimum number of uppercase alphabetic characters that must occur in the password. The maximum value must be less than or equal to the Minimum Length value. The default value is zero, which means there is no restriction of uppercase characters.
character-restriction numeric	Specifies the minimum number of numeric characters that must occur in the password. The maximum value must be less than or equal to the Minimum Length value. The default value is zero, which means there is no restriction of numeric characters.
character-restriction special-char	Specifies the minimum number of punctuation characters that must occur in the password. All printable, non-alphanumeric punctuation characters except the colon(:), exclamation mark (!), and question mark (?) are allowed. The value must be less than or equal to the Minimum Length value. The default value is zero, which means there is no restriction of punctuation characters. Special characters, such as backslash (\\) and question mark (?), are not counted as characters in a password unless the password is specified within quotes.
history	Specifies the number of old passwords against which a newly configured password is checked. The new password is discarded if it matches an old password. Range is from 0 through 10. The default is 0.
login-notify-duration	Specifies the duration in hours for which admin is notified of the number of last successful attempts. Use value 0 to disable the notification. Valid values range from is from 0 through 120. The default is 0.
min-length	Specifies the minimum length of the password. Passwords must be from 8 through 32 characters in length. The default value is 8. The total of the previous four parameters (lowercase, uppercase, digits, and punctuation) must be less than or equal to the Minimum Length value.
max-logins	Specifies the maximum number of log-in sessions allowed per local user. Range is from 0 through 10. The default is 0, representing an infinite number of log-ins.
max-retry	Specifies the number of failed password logins permitted before a user is locked out. The lockout threshold can range from 0 through 16. The default value is 0. When a password fails more than one of the strength attributes, an error is reported for only one of the attributes at a time.
repeat	Specifies the minimum number of consecutive repetitive characters in a newly configured password. The new password is discarded if it has consecutive repetitive characters (for example, aaa, xxx,1111). Configure 1 for disabling. The default is 1.
sequence	Specifies the minimum number of consecutive sequential characters both in forward and reverse direction (for example, abc, cba) in a newly configured password. The

Table 5: Password policy parameters (continued)

Parameter	Description
	new password is discarded if it has consecutive sequential characters (for example, abc, xyz, fedc). Configure 1 for disabling. The default is 1.

> **Note**
> Passwords have a maximum of 40 characters.

Password encryption policy

The software supports encrypting the passwords of all existing user accounts by enabling password encryption at the device level. By default, the encryption service is enabled.

The following rules apply to password encryption:

- When you enable password encryption, all existing clear-text passwords will be encrypted, and subsequently any passwords that are added in clear-text are stored in encrypted format.

- There are three levels of password encryption
  ◦ Encryption Level 0: No encryption, clear text.
  ◦ Encryption Level 7: AES-256 encryption.
  ◦ Encryption Level 10: SHA-512 salted HASH format. This is the default encryption level. In the following example, the testuser account password is created in clear text after password encryption has been enabled. The global encryption policy overrides command-level encryption settings, and the password is stored as encrypted.

```
device(config)# service password-encryption
device(config)# do show running-config service password-encryption
service password-encryption
device(config)# username testuser role testrole desc "Test User" encryption-level 0 password hellothere
device(config)# do show running-config username
username admin password $6$mAog0c./JxVGu1zy$6wFogQmek0KOEgTav.0DVKXz1vRodc1UCAbipYft/DWnT5R6/
Y3qpq7V3JHlhRNVtwguLgXnzdtBDKPKaXbBg/encryption-level 10 role admin desc Administrator
username testuser password $6$78rhJxmF0zFKbhu4$0WvJVdRv7.ke07E5sL7m04stPw3XO9hgIxZ/
xArDpKCPk6eGTlCn0YBi3xRv856hoiDv8U9eMxxi6ZZNY4CiV/encryption-level 10 role testrole desc "Test User"
username user password $6$mAog0c./JxVGu1zy$6wFogQmek0KOEgTav.0DVKXz1vRodc1UCAbipYft/DWnT5R6/
Y3qpq7V3JHlhRNVtwguLgXnzdtBDKPKaXbBg/encryption-level 10 role user desc User
```

> **Note**
> Clear case passwords cannot be configured with Encryption Levels 7 or 10. Clear case can only be used with Encryption Level 0.

- When you disable the password encryption service, any new passwords added in clear text will be stored as clear text on the device. Existing encrypted passwords remain encrypted.

  In the following example, the testuser account password is stored in clear text after password encryption has been disabled. The default accounts, `user` and `admin` remain encrypted.

```
device(config)# no service password-encryption
device(config)# do show running-config service password-encryption
no service password-encryption
device(config)# username testuser role testrole desc "Test User" encryption-level 0 password hellothere
enable true
device(config)# do show running-config username
```

```
username admin password $6$mAog0c./JxVGu1zy$6wFogQmek0KOEgTav.0DVKXz1vRodc1UCAbipYft/DWnT5R6/
Y3qpq7V3JHlhRNVtwguLgXnzdtBDKPKaXbBg/encryption-level 10 role admin desc Administrator
username testuser password hellothere encryption-level 0 role testrole desc "Test User"
username user password $6$mAog0c./JxVGu1zy$6wFogQmek0KOEgTav.0DVKXz1vRodc1UCAbipYft/DWnT5R6/
Y3qpq7V3JHlhRNVtwguLgXnzdtBDKPKaXbBg/encryption-level 10 role user desc User
```

- If you have passwords with encryption-level 7 on the device, then you can use the exec command `password-encryption convert-enc-to-level-10` to upgrade the passwords to encryption-level 10 (SHA-512 hash format) making the passwords more secure. Once this command is executed, all encryption-level 7 passwords are converted to encryption-level 10. However, if you downgrade to a release lower than SLX 20.1.1, these accounts will not be available.

  This command is available only to admin users. Any clear-text (encryption-level 0) passwords are retained as is in the configuration database and not converted to encryption-level 10 (SHA-512 hash format). These clear-text passwords can be converted using the `service password-encryption` configuration command.

  In the following example, testuser1 has encryption-level 7, and after running the exec command, the encryption-level is changed to 10.

```
SLX# show running-config user | inc testuser
username testuser password "cONW1RQ0nTV9Az42/9uCQg==\n" encryption-level 7 role
testrole desc "Test User"
SLX# password-encryption convert-enc-to-level-10
%WARN:This operation will convert all existing user passwords to SHA-512 format.
However, the enc level 0 (clear-text) passwords, if any, will be retained as is in the
configurationdatabase. These configurations will be lost if the system is downgraded
to lower releases than SLX 20.1.1
Do you want to continue? [Y/N]y
All passwords are converted successfully.
SLX# show running-config user | inc testuser
username testuser password $6$gV7A5lDXqcGc8/ma
$MEVxe20jaBarALGhmSYw.p3oc9IXVj9xqNUGDnfNABGs.FAqwrM8EPDMvCJcZe/MsY9geY0ej01gma7mWWWTz0
encryption-level 10 role testrole desc "Test User"
SLX#
```

  The exec command `password-encryption convert-enc-to-leve-10` is not allowed if there is a configuration rollback in-progress.

```
SLX# password-encryption convert-enc-to-level-10%WARN:This operation will convert all
existing user passwords to SHA-512 format. However, the enc level 0 (clear-text)
passwords, if any, will be retained as is in the configuration database. These
configurations will be lost if the system is downgraded to lower releases than SLX
20.1.1.
Do you want to continue? [Y/N]y
%%ERROR: Password conversion is not allowed when configuration rollback session is in
progress; Please try again later.
SLX#
```

Account lockout policy

The account lockout policy disables a user account when the user exceeds a configurable number of failed login attempts. A user whose account has been locked cannot log in. SSH login attempts that use locked user credentials are denied without the user being notified of the reason for denial.

The account remains locked until explicit administrative action is taken to unlock the account. A user account cannot be locked manually. An account that is not locked cannot be unlocked.

The account lockout policy is enforced across all user accounts except for the root account and accounts with the admin role.

Denial of service implications

The account lockout mechanism may be used to create a denial of service (DOS) condition when a user repeatedly attempts to log in to an account by using an incorrect password. Selected privileged accounts, such as root and admin, are exempted from the account lockout policy to prevent these accounts from being locked out by a DOS attack. However these privileged accounts may then become the target of password-guessing attacks.

Configuring password policies

Use the **password-attributes** command with specified parameters to define or modify existing password policies.

Configuring the account lockout threshold

You can configure the lockout threshold with the **password-attributes max-retry** *maxretry* command. The value of the *maxretry* specifies the number of times a user can attempt to log in with an incorrect password before the account is locked. The number of failed login attempts is counted from the last successful login. The *maxretry* can be set to a value from 0 through 16. A value of 0 disables the lockout mechanism (default).

The following example sets the lockout threshold to 5.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.
2. Enter the **password-attributes** command with the specified parameter.

```
device# configure terminal
Entering configuration mode terminal
device(config)# password-attributes max-retry 4
```

When a user account is locked, it can be unlocked using the procedure described in Unlocking an account on page 16.

Creating a password policy

The following example defines a password policy that places restrictions on minimum length and enforces character restrictions and account lockout.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.
2. Enter the **password-attributes** command with the specified parameters.

```
device# configure terminal
Entering configuration mode terminal
device(config)# password-attributes min-length 8 max-retry 4 character-restriction
lower 2 upper 1 numeric 1 special-char 1  max-lockout-duration 5000
```

Restoring the default password policy

Entering the **no** form of the **password-attributes** command resets all password attributes to their default values. If you specify a specific attribute, only that attribute is reset to the default. If you enter

no password-attributes without operands, all password attributes are reset to their default values.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.
2. Enter the **password-attributes** command with the specified parameters.

```
device# configure terminal
Entering configuration mode terminal
device(config)# no password-attributes min-length
device(config)# password-attributes max-retry 4
device(config)# no password-attributes numeric
```

Displaying password attributes

To display configured password attributes, change to privileged EXEC mode and enter **show running-config password-attributes**. Refer to the password-attributes command in the command reference for details on modifying password attributes.

```
device# show running-config password-attributes
password-attributes max-retry 4
password-attributes character-restriction upper 1
password-attributes character-restriction lower 2
password-attributes character-restriction numeric 1
password-attributes character-restriction special-char 1
password-attributes history 3
password-attributes login-modify-duration 1
password-attributes repeat 1
password-attributes sequence 1
```

Password interaction with remote AAA servers

The password policies apply to local device authentication only. External AAA servers such as RADIUS or TACACS+ provide server-specific password-enforcement mechanisms. The password management commands operate on the device-local password database only, even when the device is configured to use an external AAA service for authentication. When so configured, authentication through remote servers is applied to the login only.

When remote AAA server authentication is enabled, an administrator can still perform user and password management functions on the local password database.

Security-event logs

Security event logging utilizes the RASLog audit infrastructure to record security-related audit events.

Any user-initiated security event generates an auditable event.

User Accounts and Passwords Commands

The following table lists the commands that display user account and password information.

Table 6: User account and password commands in the *Command Reference*

Command	Description
`show running-config password-attributes`	Displays global password attributes.
`show running-config role`	Displays name and description of the configured roles.
`show running-config rule`	Displays configured access rules.
`show running-config username`	Displays the user accounts on the device.
`show users`	Displays the users logged in to the system and locked user accounts.
`max-logins`	Specifies the maximum number of allowed login sessions per user.

ACLs

ACL overview

An access control list (ACL) is a container for rules that permit or deny network traffic based on criteria that you specify.

When a frame or packet is received or sent, the device compares its header fields against the rules in applied ACLs. This comparison is done according to a rule sequence, which you can specify. Based on the comparison, the device either forwards or drops the frame or packet.

The benefits of ACLs include the following:

- Provide security and traffic management.
- Monitor network and user traffic.
- Save network resources by classifying traffic.
- Protect against denial of service (DOS) attacks.

Regarding the range of filtering options, there are two types of ACL:

- *Standard ACLs* — Permit, deny, or hard-drop traffic according to source address only.
- *Extended ACLs* — Permit, deny, or hard-drop traffic according to source and destination addresses, as well as other parameters. For example, in an extended ACL, you can also filter by one or more of the following:
  - Port name or number
  - Protocol, for example TCP/UDP port name or number
  - TCP flags

Regarding layer and protocol, ACL types are as follows:

- Layer 2
  - MAC ACLs

- Layer 3
  - IPv4 ACLs
  - IPv6 ACLs

For information on hardware-based filtering of IP subnet-based directed broadcast and network-address traffic, refer to "IP broadcast ACLs (bACLs)."

ACL application-targets

ACLs that you apply to interfaces or at global configuration level are summarized in a table.

You create all of these ACL types using the { `mac` | `ip` | `ipv6` } `access-list` commands.

Table 7: ACLs applied to interfaces or at global configuration level

Target/ type	Description	Applied from	Applied with	Types supported	Reference
Interface	Filters all traffic entering or exiting an interface.	Interface configuration sub-modes (including VLAN and VE)	{ `mac` | `ip` | `ipv6` } `access-group` { `in` | `out` }	MAC, IPv4, IPv6 Standard, extended	Layer 2 (MAC) ACLs on page 37 Layer 3 (IPv4 and IPv6) ACLs on page 48
Receive-path	Receive-path ACLs (rACLs) are applied at global configuration level. Their primary function is to filter CPU-bound traffic.	Global configuration mode	{ `ip` | `ipv6` } `receive access-group`	IPv4, IPv6 Standard, extended	Interface ACLs and rACLs on page 32

The following table summarizes details of ACL types not discussed in the current unit, as they differ significantly from ACLs applied to interfaces and at global configuration level.

Table 8: Other ACL applications

Target/ type	Description	Created with	Applied with	Types supported	Notes
ACL-RL	Support rate-limiting and policing; can also protect against denial of service (DOS) attacks.	{ `ip` | `ipv6` } `access-list`	`match access-group` `acl-name`	IPv4 Standard, extended	The **mirror** keyword is not supported. Applied from class-map configuration mode.
IP broadcast ACLs (bACLs)	Identify directed broadcast and network-address traffic by the specified subnets, and filter traffic on the corresponding VRF.	{ `ip` | `ipv6` } `access-list`	`ip subnet-broadcast-acl` `acl-name`	IPv4 Standard, extended	Apply the ACL at device level, interface level, or VE level.

Table 8: Other ACL applications (continued)

Target/ type	Description	Created with	Applied with	Types supported	Notes
PBR	If an incoming packet matches an ACL rule, can modify default routing behavior, for example, by changing the destination port.	`{ ip	ipv6 } access- list`	`match {ip| ipv6} address acl` `acl-name`	IPv4, IPv6 Standard, extended	The **mirror** keyword is not supported. Applied from route-map configuration mode.
Management information base (MIB)	The SNMP agent supports Get, Get-next, and Get-bulk requests for L2 ACLs on the BROCADE-ACL-MIB.	`mac access- list`	`mac access- group`	MAC Standard, extended	Refer to *Extreme SLX-OS MIB Reference.*

Interface ACLs and rACLs

Layer 3 ACLs applied at global configuration level to filter CPU-bound traffic are called *receive-path ACLs* or *rACLs*. All other ACLs discussed in this section are applied to an interface (including VLAN or VE). They can be referred to an *interface ACLs*.

Traffic entering a device can be divided into two categories:

- Datapath traffic
- CPU-bound traffic

Rules in an ACL applied to an interface filter all traffic entering or exiting that interface—datapath traffic and CPU-bound traffic.

Rules in an rACL, applied at global configuration level, primarily filter CPU-bound traffic. Implementing rACLs offers the following advantages:

- Shields the CPU from unnecessary and potentially harmful traffic.
- Mitigates denial of service (DoS) attacks.
- Protects the CPU by a single application, rather than needing to apply ACLs on multiple interfaces.

rACLs also support filtering multicast datapath traffic, which offers an alternative to applying ACLs containing multicast rules to all device interfaces.

When ACLs of multiple types are applied, processing priority is as follows: bACLs > rACLs > PBR > Layer 3 ACLs > Layer 2 ACLs. However, if any filter has a drop match, the packet is dropped irrespective of the priority.

To implement rACLs, refer to

Otherwise, continue with

ACLs applied to interfaces

This topic describes interfaces that support ACLs.

Layer 2 (MAC) ACLs are supported on the following user-interface types:

- Physical (Ethernet) interfaces—in switchport mode
- Port-channel interfaces—in switchport mode
- VLANs

Layer 3 (IPv4 and IPv6) ACLs are supported on the following interface types:

- User interfaces
  - Physical (Ethernet) interfaces
  - Port-channel interfaces
  - Virtual Ethernet (VE) (attached to a VLAN or to a bridge domain)
- Management interfaces

> **Note**
> For SLX 9150, and SLX 9250 devices, L2 and L3 ACLs applied to port-channels are supported ingress only.

ACL and rule limits

There are SW limits to the number of ACLs and rules supported.

The following software limits apply to ACL names:

- An ACL name must be unique, 1 through 63 characters long, and must begin with a–z, A–Z or 0–9. You can also use underscore (_) or hyphen (-) in an ACL name, but not as the first character.
- Rule sequence numbers can range from 1 through 65535.

The following table displays the maximum numbers of ACLs and ACL rules that you can define on a device.

Table 9: ACL and rule software limits

ACL type (standard and extended)	Maximum ACLs per type per device	Maximum rules per ACL	Maximum total rules per ACL type
Layer 2	2048	2048	102400
IPv4	2048	2048	102400
IPv6	2048	2048	102400

- Maximum of 6K ACL tables can be created per device (2K of MAC-ACL, 2K of IPv4-ACL and 2K of IPv6-ACL).

- Maximum of 300K rules can be configured per device (100K rules of MAC-ACL, 100K rules of IPv4-ACL and 100K rules of IPv6-ACL).

TCAM optimization

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

Specifying a TCAM profile and enabling TCAM sharing can help you optimize TCAM resources.

TCAM profiles

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

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

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

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

TCAM Profile Scaling (DNX Devices)

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

	Features	default	border-routing	vxlan-visibility	app-telemetry	layer2-ratelimit	multicast-profile
Ingress	L2 ACL	4K*	2K	2K	6K*	6K*	2K*
	IPv4 ACL , rACLv4, PBRv4, BGP-FLOW-SPECv4	4K*	6K*	4K	6K*	6K*	2K*
	IPv6 ACL, rACLv6, PBRv6, BGP-FLOW-SPECv6	2K	6K*	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**	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 2: TCAM-entries available per profile (DNX devices)

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

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

NS = not supported.

RL = rate limiting.

BD = bridge domain.

Default TCAM Profile (XGS Devices)

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

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

Figure 3: Maximum TCAM entries by features

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

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

NS = not supported.

RL = rate limiting.

BD = bridge domain.

Specifying a TCAM profile

Follow these steps to specify a TCAM profile.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

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

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

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

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

4. Return to privileged EXEC mode.

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

5. Save the configuration.

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

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

   ```
   device# reload system
   ```

TCAM sharing

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

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

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

Table 10: TCAM-sharing support matrix

TCAM profile	Maximum sharing-flavors	Layer 2 ACL TCAM-sharing	IPv4 ACL TCAM-sharing	IPv4 PBR TCAM-sharing	IPv6 ACL TCAM-sharing	IPv6 PBR TCAM-sharing
default	0	No	No	No	No	No
app-telemetry	0	No	No	No	No	No

Table 10: TCAM-sharing support matrix (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
border-routing	0	No	No	No	No	No
layer2-ratelimit	0	No	No	No	No	No
multicast-profile	3	Yes	Yes	Yes	No	No
vxlan-visibility	4	Yes	Yes	Yes	Yes	Yes

Enabling TCAM sharing

Follow these steps to enable TCAM sharing.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

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

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

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

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

Layer 2 (MAC) ACLs

Layer 2 access control lists (ACLs) filter traffic based on MAC header fields.

MAC ACL configuration guidelines

We present configuration guidelines for all ACLs, then for Layer 2 (MAC) ACLs.

The following guidelines are for all ACLs:

- An ACL name can be up to 63 characters long, and must begin with a–z, A–Z or 0–9. You can also use underscore (_) or hyphen (-) in an ACL name, but not as the first character.
- On any given device, an ACL name must be unique among all ACL types (MAC/IPv4/IPv6, standard or extended).
- The order of the rules in an ACL is critical. The first rule that matches the traffic stops further processing of the rules. For example, following a **permit** match, subsequent **deny** or **hard-drop** rules do not override the **permit**.
- When you create an ACL rule, you have the option of specifying the rule sequence number. If you create a rule without a sequence number, it is automatically assigned a sequence number incremented above the previous last rule.

- To modify an ACL rule, delete it and then replace it with a rule of the same **seq** number.
- You can apply a maximum of five ACLs to a user interface, as follows:
  - One ingress MAC ACL—if the interface is in switchport mode
  - One egress MAC ACL—if the interface is in switchport mode
  - One ingress IPv4 ACL
  - One egress IPv4 ACL
  - One ingress IPv6 ACL

(All supported devices) The following additional guidelines are relevant for Layer 2 ACLs:

- There is an implicit Layer 2 deny rule programmed in the CAM. This rule denies streams that do not match any of the configured rules in the ACL.
- You can apply a specific ACL to one or more interfaces, for ingress or egress, or for both.

(SLX 9540 and SLX 9640 devices) The following additional guidelines are relevant for Layer 2 ACLs:

- The **hard drop** keyword is equivalent to the **deny** keyword.
- In ingress Layer 2 ACLs, **deny** and **hard-drop** rules affect protocol packets.
- In egress Layer 2 ACLs, **deny** and **hard-drop** rules do not affect protocol packets.

(SLX 9150, and SLX 9250 devices) The following additional guidelines are relevant for Layer 2 ACLs:

- A deny match does not drop control protocol or MY IP packets .
- A hard-drop match drops all packets, including control protocol and MY IP packets.
- Layer 2 ACLs applied on VLANs do not affect tunnel-terminated packets.

Basic Layer 2 ACLs and rules

You can create standard and extended Layer 2 (MAC) ACLs, and define permit and deny rules within them.

See also Advanced Layer 2 ACL rules and features on page 41.

Creating a standard MAC ACL

A standard ACL permits or denies traffic according to source address only.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **mac access-list standard** command to create the ACL.

   ```
   device(config)# mac access-list standard test_01
   device(conf-macl-std)#
   ```

3. For each ACL rule that you need to create, enter a permit or deny command, specifying the needed parameters.

   ```
   device(conf-macl-std)# seq 100 deny host 0011.2222.3333 count
   device(conf-macl-std)# seq 110 permit host 0022.1111.2222 ffff.ffff.00ff count
   device(conf-macl-std)# deny host 0022.3333.4444 count
   device(conf-macl-std)# permit host 0022.5555.3333 count
   ```

4. Apply the ACL that you created to the appropriate interface.

Creating an extended MAC ACL

An extended ACL permits or denies traffic according to one or more of the following parameters: source address, destination address, port, ethertype, PCP value, VLAN.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **mac access-list extended** command to create the access list.

   ```
   device(config)# mac access-list extended test_02
   ```

3. For each ACL rule, enter a permit or deny command, command, specifying the needed parameters.

   ```
   device(conf-macl-ext)# seq 5 permit host 0022.3333.4444 host 0022.3333.5555
   device(conf-macl-ext)# permit host 0022.3333.5555 host 0022.3333.6666
   ```

4. Apply the ACL that you created to the appropriate interface.

Applying Layer 2 ACLs to interfaces

An ACL affects network traffic only after you apply it to an interface, using an **access-group** command. Use these procedures to apply MAC standard or extended ACLs to interfaces.

> **Note**
> MAC ACL does not filter VPLS traffic based on the VLAN that is configured in the rule.

Applying a MAC ACL to a physical interface

Use this procedure to apply a Layer 2 ACL to a physical interface in switchport mode.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command, specifying the port number.

   ```
   device(config)# interface ethernet 0/2
   ```

3. Enter the **mac access-group** command, specifying the ACL that you are applying to the interface and the in/out direction.

   ```
   device(conf-if-eth-0/2)# mac access-group test_02 in
   ```

Applying a MAC ACL to a LAG interface

Use this procedure to apply a Layer 2 ACL to a LAG (logical) interface, in switchport mode.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface port-channel** command, specifying the port-channel number.

   ```
   device(config)# interface port-channel 10
   ```

3. Enter the **mac access-group** command, specifying the ACL that you are applying to the interface and:

   - (SLX 9540 or SLX 9640 devices) The **in** or **out** direction.
   - (SLX 9150, or SLX 9250 devices) The **in** direction.
     ```
     device(config-Port-channel-10)# mac access-group test_02 in
     ```

Applying a MAC ACL to a VLAN interface

> Use this procedure to apply a Layer 2 ACL to a VLAN interface.
>
> 1. Enter the **configure terminal** command to access global configuration mode.
>
> ```
> device# configure terminal
> ```
>
> 2. Enter the **vlan** command, specifying the *vlan-id*.
>
> ```
> device(config)# vlan 50
> ```
>
> 3. Enter the **mac access-group** command, specifying the ACL that you are applying to the interface and the in/out direction.
>
> ```
> device(config-Vlan-50)# mac access-group test_02 in
> ```

Removing a MAC ACL

> To suspend ACL rules, you can remove the ACL containing those rules from the interface to which it was applied. After removing it, you can also delete the ACL.
>
> 1. Enter the **configure** command to access global configuration mode.
>
> ```
> device# configure
> ```
>
> 2. Enter the **interface** command, specifying the interface type and identifying number.
>
> ```
> device(config)# interface ethernet 2/9
> ```
>
> 3. Enter the **no access-group** command.
>
> ```
> device(conf-if-eth-2/9)# no mac access-group macacl2 in
> ```

Layer 2 ACL modification

> You can replace the contents of an ACL rule. You can also modify ACL sequence (**seq**) numbers.

Modifying MAC ACL rules

> To modify an ACL rule, delete the original rule and replace it with a new rule.
>
> 1. To display MAC ACL rule details, in privileged EXEC mode enter the **show running-config mac access-list** command.
>
> ```
> device# show running-config mac access-list standard ACL1
> mac access-list standard ACL1
>   seq 100 deny host 0022.3333.4444 count
>   seq 110 permit host 0011.3333.5555 count
> ```
>
> Note the **seq** number of the rule that you need to modify.
>
> 2. Enter the **configure terminal** command to access global configuration mode.
>
> ```
> device# configure terminal
> ```
>
> 3. Enter the **mac access-list** command, specifying the ACL you need to modify.
>
> ```
> device(config)# mac access-list standard ACL1
> ```

4. Delete the original rule, doing one of the following:

- Enter the **no seq** command, specifying the sequence number of the rule that you are deleting.

```
device(conf-macl-std)# no seq 100
```

- Enter the exact rule that you are deleting, preceded by **no**.

```
no deny host 0022.3333.4444 count
```

5. Enter the replacement rule.

```
device(conf-macl-ext)# seq 100 permit host 0022.3333.6666 count
```

Reordering the sequence numbers in a MAC ACL

Reordering ACL-rule sequence numbers is helpful if you need to insert new rules into an ACL in which there are not enough available sequence numbers.

Note the following regarding sequence numbers and their reordering parameters:

- The default initial sequence number is 10 and the default increment is 10.
- For reordering the sequence numbers, you need to specify the following:
  - The new starting sequence number
  - The increment between sequence numbers

The first rule receives the number of the starting sequence number that you specify. Each subsequent rule receives a number larger than the preceding rule. The difference in numbers is determined by the increment number that you specify. The starting-sequence number can range from 1 through 65535, and the increment number can range from 1 through 65534.

For example: In the command below, the **resequence access-list** command assigns a sequence number of 50 to the first rule, 55 to the second rule, 60 to the third rule, and so forth.

```
device# resequence access-list mac test_02 50 5
```

Advanced Layer 2 ACL rules and features

Many advanced ACL features are implemented per ACL rule, according to parameters that you specify.

> **Note**
> Some advanced features also require global configuration.

Table 11: Layer 2 ACL advanced keywords

Keyword	Description	L2 standard ACL	L2 extended ACL	Notes
copy-sflow	sFlow monitoring	P/D/H; I	P/D/H; I	
count	Counter statistics	P/D/H; I/O	P/D/H; I/O	

Table 11: Layer 2 ACL advanced keywords (continued)

Keyword	Description	L2 standard ACL	L2 extended ACL	Notes
drop-precedence-force	Re-marking drop-precedence	NA	P; I	Supported only under **default**, **vxlan-visibility**, or **border-routing** TCAM profiles.
log	Logging	P/D/H; I	P/D/H; I	
mirror	Mirroring	NA	P/D/H; I	
pcp	802.1p filtering	NA	P/D/H; I/O	
pcp-force	802.1p re-marking	NA	P; I	
vlan-tag-format	Filtering by **untagged**, **single-tagged**, or **double-tagged** VLAN type	NA	P/D/H; I/O	The **vlan-tag-format** keyword is supported on the SLX-9540/9640 in the Layer2-Ratelimit profile only. Also, when multi-tagged packets are sent (packets with more than 2 tags) the rule written for **vlan-tag-format double-tagged** is matched; DNX BCM HW assumes the multi-tag packet to be double-tagged. The **vlan-tag-format** keyword is not supported on the SLX-9150/9250.

Key:

- **P**—Supported in a permit rule.
- **D**—Supported in a deny rule.
- **H**—Supported in a hard-drop rule. For Layer 2 ACLs on SLX 9540 and SLX 9640 devices, the **hard-drop** keyword functions exactly like the **deny** keyword. For other devices, **hard-drop** overrides the trap behavior for control frames.
- **I**—Supported in an ACL applied to incoming traffic.
- **O**—Supported in an ACL applied to outgoing traffic.
- **NA**—Not available.

For details, refer to the following *Extreme SLX-OS Command Reference* topics:

- seq (rules in MAC standard ACLs)
- seq (rules in MAC extended ACLs)

Parsing priorities among keywords

There are parsing priorities among the **copy-sflow**, **log**, and **mirror** keywords, as follows:

- Although in a standard-ACL rule you can include **log** and **copy-sflow**, only one of the two is processed, as follows:
  - In a permit rule, the order of precedence is **copy-sflow** > **log**.
  - In a deny or hard-drop rule, the order of precedence is **log** > **copy-sflow**.
- Although in an extended-ACL rule you can include **log**, **mirror**, and **copy-sflow**, only one of the three is processed, as follows:
  - In a permit rule, the order of precedence is **mirror** > **copy-sflow** > **log**.
  - In a deny or hard-drop rule, the order of precedence is **log** > **copy-sflow** > **mirror**.

Consider the following extended Layer 2 ACL:

```
device(config)# mac access-list extended mac1
device(conf-macl-ext)# seq 10 permit host 0000.1324.3333 any count log mirror copy-sflow
device(conf-macl-ext)# seq 20 deny host 0000.1357.4444 any count log mirror copy-sflow
```

- In the permit rule, only the **mirror** keyword is processed.
- In the deny rule, only the **log** keyword is processed.

Creating MAC ACL rules enabled for counter statistics

When you create ACL rules, the **count** parameter enables you to display counter statistics.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **mac access-list** command to create or modify an access list.

   ```
   device(config)# mac access-list standard mac_acl_1
   ```

3. In each rule for which you need to display statistics, include the **count** keyword.

   ```
   device(conf-mac1-std)# seq 100 deny 0022.3333.4444 count
   ```

4. If you have not yet applied the ACL to the appropriate interface, do so now.
5. (Optional) To display ACL counter statistics, enter the **show statistics access-list** command.

Filtering by VLAN tag type (L2 ACLs)

In Layer 2 extended-ACL rules, you can filter ingress traffic by untagged, single-tagged, or double-tagged VLAN type.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **mac access-list extended** command to create or access the ACL.

   ```
   device(config)# mac access-list extended mac_ac13
   ```

3. To filter by untagged VLANs, create rules including the **vlan-tag-format untagged** parameters.

```
device(conf-macl-ext)# permit host 0001.0001.0001 any vlan-tag-format untagged vlan
100 count
device(conf-macl-ext)# permit host 0001.0001.0004 any vlan-tag-format untagged vlan
100 count
```

4. To filter by single-tagged VLANs, create rules including the **vlan-tag-format single-tagged** parameters.

```
device(conf-macl-ext)# permit host 0002.0002.0002 any vlan-tag-format single-tagged
vlan 200 count
device(conf-macl-ext)# deny host 1.2.3 any vlan-tag-format single-tagged vlan 101
0xff0 count
```

5. To filter by double-tagged VLANs, create rules including the **vlan-tag-format double-tagged** parameters.

```
device(conf-macl-ext)# permit host 0003.0003.0003 any vlan-tag-format double-tagged
outer-vlan 300 inner-vlan 400 count
device(conf-macl-ext)# permit host 0003.0003.0005 any vlan-tag-format double-tagged
outer-vlan 300 0xfff inner-vlan 400 0x0fff count
device(conf-macl-ext)# permit host 0003.0003.0006 any vlan-tag-format double-tagged
outer-vlan any inner-vlan any count
```

6. Apply the ACL to the appropriate interface.

```
device(conf-macl-ext)# exit
device(config)# interface ethernet 2/1
device(conf-if-eth-2/1)# mac access-group mac_acl3 in
```

Filter and Force PCP Values

In Layer 2 extended ACL rules, re-marking (forcing) PCP values can change the priority on ingress traffic.

You can also filter ingress and egress Layer 2 packets by PCP value.

1. Access access global configuration mode.

```
device# configure
```

2. Create or access the ACL.

```
device(config)# mac access-list extended mac_ac12
```

3. To filter incoming or outgoing packets by PCP value, use the **pcp** parameter to define permit and deny rules.

```
device(conf-macl-ext)# seq 5 permit host 0022.3333.4444 host 0022.3333.5555 pcp 2
device(conf-macl-ext)# deny host 0022.3333.7777 host 0022.3333.6666 pcp 5
```

4. To re-mark the PCP value of incoming packets, use the **pcp-force** parameter to define permit rules.

```
device(conf-macl-ext)# seq 10 permit host 0022.3333.4445 host 0022.3333.5556 pcp-force
2
```

| Note
| The **pcp-force** parameter is not supported for the SLX 9740.

5. Apply the ACL to the appropriate interface.

```
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# mac access-group mac_acl2 in
```

Filtering by known-unicast-only and unknown-unicast-only

Configure extended ACL rules to filter traffic by know-unicast-only and unknown-unicast-only.

This procedure configures filtering for known or unknown unicast traffic only. It is supported on Ingress ACL only on the L2 ratelimit profile. Implict deny will be applied on both known and unknown unicast traffic.

1. Enter **configure** to access global configuration mode.

   ```
   device# configure
   ```

2. Enter the **mac access-list extended** command to create or access the ACL.

   ```
   device(config)# mac access-list extended mac_ext22
   ```

3. Configure filtering on the ACL for known or unknown unicast traffic.

   ```
   device(conf-macl-ext)# permit any any known-unicast-only
   ```

4. Apply the ACL to the appropriate interface.

   ```
   device(config)# interface ethernet 0/1
   device(conf-if-eth-0/1)# ip address mac_ext22 in
   ```

ACL logs

ACL logs can provide insight into permitted and denied network traffic.

ACL logs maintain the following properties:
- Supported for all ACL types (MAC, IPv4, and IPv6)
- Supported for incoming network traffic only
- Supported for all user interfaces (but not on management interfaces) on which ACLs can be applied
- May be CPU-intensive

You can also enable Raslog logging for ACLs.

Enabling and configuring the ACL log buffer

Among the conditions required for ACL logging is that the ACL log buffer be enabled and configured.

1. Enter the **debug access-list-log buffer** command to enable and configure ACL log buffering.

   ```
   device# debug access-list-log buffer circular packet count 1600
   ```

2. (Optional) To display the current ACL log buffer configuration, enter the **show access-list-log buffer config** command.

   ```
   device# show access-list-log buffer config
   ACL Logging is enabled
   Buffer exists for interface Eth 1/11
   Buffer type is Circular and size is 1000
   ```

Creating a MAC ACL rule enabled for logging

When you create ACL rules for which you want to enable logging, you must include the **log** keyword.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **mac access-list** command to create or modify an access list.

```
device(config)# mac access-list standard mac_1
```

3. In each rule for which you need logging, include the **log** keyword.

```
device(conf-mac1-std)# seq 100 deny 0022.3333.4444 log
```

4. If you have not yet applied the ACL to the appropriate interface, do so now.

   To enable logging for Layer 2 implicit deny rules, use the command **implicit-deny-log l2acl**.

   > **Note**
   > The **implicit-deny-log l2acl** is under `acl-policy`. After using **implicit-deny-log l2acl**, the user must rebind the L2 ACL to ensure the change takes effect.

5. (Optional) To display ACL logs, enter the **show access-list log buffer** command.

Enabling and configuring ACL Raslogs

This task enables Raslog messages for ACL rules with **log** keywords and specifies how long the system waits between ACL Raslog messages.

For details of Raslog messages, refer to the *Extreme SLX-OS Message Reference*.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **acl-policy** command to access ACL policy configuration mode.

```
device(config)# acl-policy
```

3. Enter the **acl-log-raslog** command to enable ACL Raslogs.

```
device(config-acl-policy)# acl-log-raslog
```

4. To modify the interval between the first ACL Raslog and each consecutive Raslog, enter the **acl-log-raslog log-interval** command.

```
device(config-acl-policy)# acl-log-raslog log-interval 8
```

The following output is an ACL Raslog example.

```
MAC ACL mac_2 permitted 1 packets on intf eth1/6 [SA:0010.1010.1001, DA:0001.0300.0500,
Type:0, VLAN:101, SIP:0.0.0.0, DIP:0.0.0.0, l3_proto:none, src_port:0, dst_port:0]

IP ACL v4acl denied 1 packets on intf eth1/6  [SA:0001.0300.0400,DA:0001.0300.0500,
Type:800, VLAN:100, SIP:2.2.2.2, DIP:6.6.6.6, l3_proto:udp, src_port:66, dst_port:77]

IPv6 ACL v6acl permitted 1 packets on intf po44  [SA:0001.0300.0400,DA:0001.0300.0500,
Type:86dd, VLAN:100, SIP:fe80::201:3ff:fe00:400,
DIP:3555:5555:6666:6666:7777:7777:8888:8888,
l3_proto:udp, src_port:63, dst_port:63]
```

Layer 2 ACL-based mirroring

ACL-based mirroring enables you to monitor specified inbound traffic in the mirrored port by attaching a protocol analyzer to the mirror.

ACL mirroring is supported for *ethernet*, *port-channels*, and *ve* ingress interfaces.

Enabling L2 ACL rules for mirroring

ACL-based inbound mirroring applies to extended-ACL rules that include the **mirror** keyword.

1. Enter **configure** to access global configuration mode.

```
device# configure
```

2. Enter the **mac access-list extended** command to create or access the ACL.

```
device(config)# mac access-list extended mac_ac12
```

3. In each rule for which you need to enable mirroring, include the **mirror** keyword.

```
device(conf-macl-ext)# seq 5 permit host 0022.3333.4444 host 0022.3333.5555 mirror
device(conf-macl-ext)# deny host 0022.3333.7777 host 0022.3333.6666 mirror
```

4. Apply the ACL that you created to the appropriate physical interface, specifying the **in** keyword.

```
device(config)# interface ethernet 2/1
device(conf-if-eth-2/1)# mac access-group mac_acl2 in
```

Defining an ACL mirror port

To support ACL mirroring, define a destination ACL mirror port common to all ports of a port processor (PPCR). ACL mirroring is supported for *ethernet*, *port-channels*, and *ve* ingress interfaces.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. To define a physical interface as the mirror, enter a command such as the following example.

```
device(config)# acl-mirror source ethernet 0/1 destination ethernet 0/2
```

3. To define a port-channel as the mirror, enter a command such as the following example.

```
device(config)# acl-mirror source ethernet 0/1 destination port-channel 1
```

4. To define a port-channel as the source for the mirror, enter a command such as the following example.

```
device(config)# acl-mirror source port-channel 3 destination port-channel 6
```

Enabling L2 ACL rules for sFlow monitoring

sFlow is a sampling technology for monitoring networks. You can monitor specified incoming data flows by including the **copy-sflow** keyword in rules within an ACL applied to a device.

In order for sFlow to function, the sFlow collector must be globally configured. For details, refer to the *Extreme SLX-OS Monitoring Configuration Guide* .

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **mac access-list standard/extended** command to create or access the ACL.

```
device(config)# mac access-list extended mac_ac12
```

3. In each rule for which you need to enable sFlow, include the **copy-sflow** keyword.

```
device(conf-macl-ext)# seq 5 permit host 0022.3333.4444 host 0022.3333.5555 copy-sflow
device(conf-macl-ext)# deny host 0022.3333.7777 host 0022.3333.6666 copy-sflow
```

4. Apply the ACL that you created to the appropriate physical interface, specifying **in**.

```
device(config)# interface ethernet 2/1
device(conf-if-eth-2/1)# mac access-group mac_acl2 in
```

ACL show and clear commands

There is a full range of ACL show and clear commands, listed here with descriptions.

Table 12: ACL show commands in the *Command Reference*

Command	Description
`show access-list`	For a given network protocol and inbound/outbound direction, displays ACL information. You can show information for a specified ACL or only for that ACL on a specified interface. You can also display information for all ACLs bound to a specified interface.
`show access-list-log buffer`	Displays the contents of the ACL log buffer.
`show access-list-log buffer config`	Displays the ACL log buffer configuration.
`show running-config {mac	ip	ipv6} access-list`	For a given network protocol and standard/extended type, displays ACL configuration. You can show the configuration of a specified ACL or for all such ACLs.
`show statistics access-list`	For a given network protocol and inbound/outbound direction, displays statistical information—for ACL rules that include the **count** keyword. You can show statistics for a specified ACL or only for that ACL on a specified interface. You can also display statistical information for all ACLs bound to a specified interface or to the management interface.

Table 13: ACL clear commands in the *Command Reference*

Command	Description
`clear counters access-list`	For a given network protocol and inbound/outbound direction, clears ACL statistical information. You can clear all statistics for a specified ACL or only for that ACL on a specified interface. You can also clear statistical information for all ACLs bound to a specified interface or to the management interface.

Layer 3 (IPv4 and IPv6) ACLs

Layer 3 access control lists (ACLs) filter traffic based on IPv4 or IPv6 header fields.

Implementation flows for rACLs and interface ACLs

The implementation flows for Layer 3 interface ACLs and receive-path ACLs (rACLs) are similar.

> **Note**
> For a comparison of rACLs and interface ACLs, refer to Interface ACLs and rACLs on page 32.

The following table displays the differential flows of implementation topics for Layer 3 interface ACLs and rACLs:

Table 14: Interface ACLs and receive-path ACLs

IPv4/IPv6 interface ACLs	All IPv4/IPv6 ACLs	IPv4/IPv6 rACLs
	Layer 3 ACL configuration guidelines on page 49	
	One of the following procedures: • Creating a standard IPv4 ACL on page 53 • Creating a standard IPv6 ACL on page 53 • Creating an extended IPv4 ACL on page 54 • Creating an extended IPv6 ACL on page 54	
Applying Layer 3 ACLs to interfaces or globally on page 55		Applying rACLs to devices on page 58

The above table indicates that there are no structural differences between Layer 3 interface ACLs and rACLs; you use identical procedures for all types. The implementation differences are as follows:

- You apply interface ACLs from an interface (including VE) configuration mode, using the { `ip | ipv6` } `access-group` command.

- You apply rACLs from global configuration mode, using the { `ip | ipv6` } `receive access-group` command.

- Mirroring is not supported for rACLs.

Layer 3 ACL configuration guidelines

We present configuration guidelines for all ACLs, then for Layer 3 ACLs, then for L3 ACLs applied to a user interface, then for ACLs applied to the management interface, and then guidelines for receive-path ACLs (rACLs).

The following guidelines are for all ACLs:

- An ACL name can be up to 63 characters long, and must begin with a–z, A–Z or 0–9. You can also use underscore (_) or hyphen (-) in an ACL name, but not as the first character.

- On any given device, an ACL name must be unique among all ACL types (MAC/IPv4/IPv6, standard or extended).

- The order of the rules in an ACL is critical. The first rule that matches the traffic stops further processing of the rules. For example, following a **permit** match, subsequent **deny** or **hard-drop** rules do not override the **permit**.
- When you create an ACL rule, you have the option of specifying the rule sequence number. If you create a rule without a sequence number, it is automatically assigned a sequence number incremented above the previous last rule.
- To modify an ACL rule, delete it and then replace it with a rule of the same **seq** number.
- You can apply a maximum of five ACLs to a user interface, as follows:
  - One ingress MAC ACL—if the interface is in switchport mode
  - One egress MAC ACL—if the interface is in switchport mode
  - One ingress IPv4 ACL
  - One egress IPv4 ACL
  - One ingress IPv6 ACL

Guidelines for all Layer 3 ACLs

(All supported devices) In addition to the guidelines that apply to all ACLs, the following guidelines are relevant for Layer 3 ACLs:

- In ingress Layer 3 ACLs, **hard-drop** rules affect control protocol and MY IP packets.
- On Management interfaces, a **hard-drop** works the same as a **deny**action.
- Although L3 ACL **deny** rules do not drop protocol packets, best practice is to define an explicit permit rule for needed protocols.

(SLX 9150, and SLX 9250 devices) The following additional guidelines are relevant for Layer 3 ACLs:

- For tunnel-termination cases, ingress ACLs are applicable for inner headers.
- For tunnel-origination cases, egress ACLs are applicable for outer headers.

(SLX 9540 and SLX 9640 devices) The following additional guidelines are relevant for Layer 3 ACLs:

- In egress Layer 3 ACLs, **deny** and **hard-drop** rules do not affect control protocol and MY IP packets.

Guidelines for Layer 3 ACLs applied to user interfaces

(All supported devices) In addition to the previous guidelines, the following guidelines are relevant for Layer 3 ACLs applied to user interfaces:

- There is an implicit "deny" rule at the end of every Layer 3 ACL applied to a user interface. This denies all L3 streams that do not match any of the configured rules in the ACL.
- Traffic generated by the CPU—for example, echo request packets—are not filtered by egress IPv4 ACLs.

(SLX 9540 and SLX 9640 devices) In addition to the previous guidelines, the following guidelines are relevant for Layer 3 ACLs applied to user interfaces:

- Egress IPv4 ACLs are applicable only for routed traffic.

Guidelines for ACLs applied to the management interface

The following protection guards against malicious ICMP timestamp requests (icmp-type 13):

- ICMP timestamp requests and responses are dropped by default.

- If an ACL with a **permit icmp any any** rule is applied to the management interface, such a rule permits ICMP timestamp requests. However, ICMP timestamp responses are blocked.

The following additional guidelines are relevant for Layer 3 ACLs applied to the management interface:

- When an ACL is bound to the management interface, ICMP packet types are implicitly permitted. An implicit deny entry is programmed for the rest of the non-matching traffic.
- (Standard ACLs) Only packets with TCP/UDP protocols are filtered for the configured match condition (for example, SIP).
- (Standard ACLs) By default, TCP, UDP, ESP, AH, and ICMP are allowed.
- (Extended ACLs) Applying a permit or deny UDP ACL to the management interface enacts an implicit deny for TCP; however, a ping will succeed.
- (Extended ACLs) Applying a permit or deny ACL for a specific UDP port enacts an implicit deny for all other UDP ports.
- (Extended ACLs) Applying a permit or deny ACL for a specific TCP port enacts an implicit deny for all other TCP ports.
- You can apply a maximum of two ACLs to the management interface, as follows:
  - One ingress IPv4 ACL
  - One ingress IPv6 ACL
- Before downgrading firmware, unbind any ACLs on the management interface.

If no ACLs are applied to the device management interface, ICMP pings are allowed. In addition, the following default rules are effective:

- seq 0 permit tcp any any eq 22
- seq 1 permit tcp any any eq 23
- seq 2 permit tcp any any eq 80
- seq 3 permit tcp any any eq 443
- seq 4 permit udp any any eq 161
- seq 5 permit udp any any eq 123
- seq 6 permit tcp any any range 600-65535
- seq 7 permit udp any any range 600-65535

Guidelines for rACLs

(All supported devices) The following additional guidelines are relevant for all receive-path ACLs (rACLs):

- Interface ACLs and rACLs share the same resource (database-table).
- In all rACLs, explicit and implicit rules are processed in the following order:

  1. Explicit rules, in an order determined by their **seq** numbers.
  2. An implicit **deny any** *my-ip* rule that affects all other CPU-bound traffic.

- Under inband management, you need to include permit rules for your telnet/SSH access to the device.

(SLX 9150, and SLX 9250 devices) The following additional guidelines are relevant for all receive-path ACLs (rACLs):

- IPv4 rACLs apply to multicast datapath traffic only if multicast destination-IPs are explicitly specified in rules.
- In an IPv4 rACL rule, if a destination IP is not specified, $my-ip$ (IP addresses configured on any Layer 3 interface) is interpreted as the destination IP. Such rules do not filter multicast traffic.
- Multicast traffic is first filtered by rACLs, then by interface ACLs.

(SLX 9540 and SLX 9640 devices) The following additional guidelines are relevant for all receive-path ACLs (rACLs):

- rACLS are supported only for unicast, routed traffic.
- In an IPv4 rACL rule, if a destination IP is not specified, matches apply to all IP addresses configured on the device.

(All supported devices) The following guidelines are relevant for multiple rACLs:

- Multiple rACLs are supported only in the default TCAM profile.
- You can apply a maximum of 400 receive-path ACLs to a device, as follows:
  - 200 IPv4 receive-path ACLs
  - 200 IPv6 receive-path ACLs
- Supported sequence numbers for rACLs range from 1 through 2047.
- If you apply an rACL to a device without specifying a sequence number, numbers are assigned automatically, as follows:
  - The first rACL applied to the device is assigned 10.
  - Each additional rACL applied is assigned an increment of 10 above the highest applied sequence number.

The following table compares the effect of **deny** or **hard-drop** rACL matches for different types of packets.

Table 15: Effect of deny and hard-drop matches on different packet types

Packet type	Deny or hard-drop match
Control packets	Not dropped
Data packets	NA
My IP packets (incl. ICMP, control or data)	Dropped

Basic Layer 3 ACLs and rules

You can create standard and extended Layer 3 (IPv4 and IPv6) ACLs, and define permit and deny rules within them.

ACLs now support filtering fragmented and non-fragmented packets. Use the `fragment` or `non-fragment` keywords to enable filtering out these packets. Filtering on *fragment* and *non-fragment* packets is not supported in SLX 9150 and SLX 9250.

See also Advanced Layer 3 ACL rules and features on page 61.

Creating a standard IPv4 ACL

A standard ACL permits or denies traffic according to source address only.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **ip access-list standard** command to create the access list.

   ```
   device(config)# ip access-list standard stdACL3
   ```

3. Enter rules, specifying the needed parameters.

   ```
   device(conf-ipacl-std)# seq 5 permit host 10.20.33.4
   device(conf-ipacl-std)# seq 15 deny any
   ```

4. Apply the ACL that you created to the appropriate interface.

The following example shows how to create a standard IPv4 ACL, define rules for it, and apply the ACL to an interface.

```
device# configure
device(config)# ip access-list standard stdACL3
device(conf-ipacl-std)# seq 5 permit host 10.20.33.4
device(conf-ipacl-std)# seq 10 permit 20.20.33.5
device(conf-ipacl-std)# seq 15 deny any
device(conf-ipacl-std)# exit
device(config)# interface ethernet 5/2
device(conf-if-eth-5/2)# ip access-group stdACL3 in
```

Creating a standard IPv6 ACL

A standard ACL permits or denies traffic according to source address only.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 access-list standard** command to create the access list.

   ```
   device(config)# ipv6 access-list standard std_V6_ACL4
   ```

3. Enter rules, specifying the needed parameters.

   ```
   device(conf-ip6acl-std)# seq 5 permit host 2001:db8::1:2
   device(conf-ip6acl-std)# seq 15 deny any
   ```

4. Apply the ACL to the appropriate interface.

Creating an extended IPv4 ACL

An extended ACL permits or denies traffic according to one or more parameters, including source address, destination address, port, protocol (TCP or UDP), and TCP flags.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ip access-list extended** command to create the access list.

```
device(config)# ip access-list extended extdACL5
```

3. Enter rules, specifying the needed parameters.

```
device(conf-ipacl-ext)# seq 5 deny tcp host 10.24.26.145 any eq 23
device(conf-ipacl-ext)# seq 7 deny tcp any any eq 80
device(conf-ipacl-ext)# seq 10 deny udp any any range 10 25
device(conf-ipacl-ext)# seq 15 permit tcp any any
```

4. Apply the ACL to the appropriate interface.

The following example creates an IPv4 extended ACL, defines rules in the ACL, and applies it as a receive-path ACL.

```
device(config)# ip access-list extended ipv4-receive-acl-example
device(conf-ipacl-ext)# deny tcp host 10.0.0.1 any count
device(conf-ipacl-ext)# deny udp any host 20.0.0.1 count
device(conf-ipacl-ext)# permit tcp host 10.0.0.2 any eq telnet count
device(conf-ipacl-ext)# permit tcp host 10.0.0.2 any eq bgp count
device(conf-ipacl-ext)# deny tcp host 10.0.0.3 host 224.0.0.1 count

device(conf-ipacl-ext)# exit
device(config)# ip receive access-group ipv4-receive-acl-example
```

Creating an extended IPv6 ACL

An extended ACL permits or denies traffic according to one or more parameters, including source address, port, protocol (TCP or UDP), and TCP flags.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ipv6 access-list extended** command to create the access list.

```
device(config)# ipv6 access-list extended ipv6_acl_1
```

3. Enter rules, specifying the needed parameters.

```
device(conf-ip6acl-ext)# seq 10 deny ipv6 2001:2002:1234:1::/64 2001:1001:1234:1::/64
count
```

4. Apply the ACL to the appropriate interface, specifying the **in** direction.

```
device(conf-ip6acl-ext)# exit
device(config)# interface ethernet 0/22
device(conf-if-eth-0/22)# ipv6 access-group ipv6_acl_1 in
```

The following example shows how to create an extended IPv6 ACL, define rules for it (including a rule that filters by DSCP ID), and apply the ACL to an interface.

```
device# configure terminal
device(config)# ipv6 access-list extended ip_acl_1
device(conf-ip6acl-ext)# seq 10 deny ipv6 any any dscp 3
device(conf-ip6acl-ext)# seq 20 deny ipv6 2001:2002:1234:1::/64 2001:1001:1234:1::/64
count
device(conf-ip6acl-ext)# exit
```

```
device(config)# interface ethernet 0/22
device(conf-if-eth-0/22)# ipv6 access-group ipv6_acl_1 in
```

The following example creates an IPv6 extended ACL, defines rules in the ACL, and applies it as a receive-path ACL.

```
device(config)# ipv6 access-list extended ipv6-receive-acl-example
device(conf-ipacl-ext)# hard-drop tcp host 10::1 any count
device(conf-ipacl-ext)# hard-drop udp any host 20::1 count
device(conf-ipacl-ext)# permit tcp host 10::2 any eq telnet count
device(conf-ipacl-ext)# permit tcp host 10::2 any eq bgp count
device(conf-ipacl-ext)# hard-drop tcp host 10::3 host ff02::1 count

device(conf-ipacl-ext)# exit
device(config)# ipv6 receive access-group ipv6-receive-acl-example
```

Applying Layer 3 ACLs to interfaces or globally

An ACL affects network traffic only after you apply it to an interface or globally, using one of the **access-group** commands. Use these procedures to apply standard or extended IPv4 and IPv6 ACLs or to remove them.

Applying a Layer 3 ACL to a physical interface

Use this procedure for applying an IPv4 or IPv6 ACL to a physical interface, using the **ip/ipv6 access-group** command.

1. Enter **configure terminal** to change to global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command, specifying the port number.

   ```
   device(config)# interface ethernet 0/2
   ```

3. Enter the **ip/ipv6 access-group** command, specifying the ACL that you are applying to the interface.

   - For IPv4 ACLs, specify the ingress or egress direction.
     ```
     device(conf-if-eth-0/2)# ip access-group test_02 out
     ```

   - For IPv6 ACLs, specify the ingress direction.
     ```
     device(conf-if-eth-0/2)# ipv6 access-group stdV6ACL_1 in
     ```

4. Enter the **ip/ipv6 access-group** command, specifying the ACL that you are applying to the interface.

   - For IPv4 ACLs, specify the ingress or egress direction.
     ```
     device(conf-if-eth-0/2)# ip access-group test_02 out
     ```

   - For IPv6 ACLs, specify the ingress direction.
     ```
     device(conf-if-eth-0/2)# ipv6 access-group stdV6ACL_1 in
     ```

The following example applies an IPv4 ACL to a physical interface.
```
device# configure
device(config)# interface ethernet 0/2
device(conf-if-eth-0/9)# ip access-group ipacl2 in
```

The following example applies an IPv6 ACL to a physical interface.
```
device# configure
device(config)# interface ethernet 0/2
```

```
device(conf-if-eth-0/2)# ipv6 access-group ip_acl_1 in

device(conf-if-eth-0/2)# do show access-list ipv6 ip_acl_1 in
ipv6 access-list ip_acl_1 on ethernet 0/22 at Ingress (From User)
    seq 10 deny ipv6 2001:2002:1234:1::/64 2001:1001:1234:1::/64 count (Active)
```

Applying a Layer 3 ACL to a LAG interface

Use this procedure to apply an IPv4 or IPv6 ACL to a LAG (logical) interface.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface port-channel** command, specifying the port-channel number.

   ```
   device(config)# interface port-channel 10
   ```

3. Enter the **ip/ipv6 access-group** command, specifying the ACL that you are applying to the interface.

   - For IPv4 ACLs, specify the ingress or egress direction.
     ```
     device(config-Port-channel-10)# ip access-group test_02 out
     ```

   - (SLX 9150, or SLX 9250 devices) For IPv6 ACLs, specify the ingress direction.
     ```
     device(config-Port-channel-10)# ipv6 access-group stdV6ACL_1 in
     ```

   - (SLX 9540 or SLX 9640 devices) For IPv6 ACLs, specify the ingress or egress direction.
     ```
     device(config-Port-channel-10)# ipv6 access-group stdV6ACL_1 out
     ```

Applying a Layer 3 ACL to a VE interface

Use this procedure to apply an IPv4 or IPv6 ACL to a VE interface (attached to a VLAN).

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface ve** command, specifying the *vlan-id*.

   ```
   device(config)# interface ve 50
   ```

3. Enter the **ip/ipv6 access-group** command, specifying the ACL that you are applying to the VE.

   - For IPv4 ACLs, specify the ingress or egress direction.
     ```
     device(config-if-ve-50)# ip access-group test_02 out
     ```

   - For IPv6 ACLs, specify the ingress direction.
     ```
     device(config-if-ve-50)# ipv6 access-group stdV6ACL_1 in
     ```

Applying a Layer 3 ACL to a VE interface (bridge-domain)

Use this procedure to apply an IPv4 or IPv6 ACL to a VE interface (attached to a bridge-domain).

> **Note**
> For details of VE on bridge-domain, refer to *Extreme SLX-OS Layer 2 Switching Configuration Guide*.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Make sure that the Layer 3 ACL you require is defined.

- IPv4

```
device(config)# ip access-list extended ipv4_1
device(conf-ipacl-ext)#  permit ip host 1.1.1.1 any count
device(conf-ipacl-ext)# exit
```

- IPv6

```
device(config)# ipv6 access-list standard stdV6ACL_1
device(conf-ipv6-std)# seq 10 permit 2001:db8:85a3:0:0:8a2e:370:7334
device(conf-ipv6-std)# seq 11 deny any
device(conf-ipv6-std)# exit
```

3. Make sure that a non-default pseudowire profile is defined, with tagging mode enabled.

```
device(config)# pw-profile temp
device(config-pw-profile-temp)# vc-mode tag
device(config-pw-profile-temp)# exit
```

4. Make sure that a logical interface is defined on a physical interface.

```
device(config)# interface ethernet 0/7
device(conf-if-eth-1/3)# switchport
device(conf-if-eth-1/3)# switchport mode trunk
device(conf-if-eth-1/3)# logical-interface ethernet 0/7.100
device(conf-if-eth-lif-0/7.100)# vlan 100
```

5. Make sure that the bridge-domain you require is configured, with a VE attached as router interface.

```
device(config)# bridge-domain 10
device(config-bridge-domain-10)# pw-profile temp
device(config-bridge-domain-10)# vc-id 10
device(config-bridge-domain-10)# peer 12.12.12.12
device(config-bridge-domain-10)# router-interface Ve 10
```

6. Access the VE attached to the bridge-domain.

```
device(config-bridge-domain-10)# exit
device(config)# interface ve 10
```

7. Enter the **ip/ipv6 access-group** command, specifying the ACL that you are applying to the VE.

- For IPv4 ACLs, specify the ingress or egress direction.
  ```
  device(config-if-Ve-10)# ip access-group ipv4_1 out
  ```

- For IPv6 ACLs, specify the ingress direction.
  ```
  device(config-if-Ve-10)# ipv6 access-group stdV6ACL_1 in
  ```

8. Enter the **no shutdown** command.

```
device(config-if-Ve-10)# no shutdown
```

Applying a Layer 3 ACL to the management interface

Use this procedure for applying a Layer 3 ACL to the management interface, using the **{ip | ipv6} access-group** command.

> **Note**
> If an explicit "deny ip any any" IP rule is applied to the management interface, that IP rule has priority over any TCP or UDP rules. Any incoming TCP packets that match that IP rule are dropped because the TCP packet has an IP header.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Use the **interface management** command to enter configuration mode for the management interface.

   ```
   device(config)# interface management 0
   ```

3. To apply an IPv4 ACL to the management interface, enter the **ip access-group** command, specifying the ACL that you are applying to the interface, and **in**.

   ```
   device(config-Management-0)# ip access-group stdACL3 in
   ```

4. To apply an IPv6 ACL to the management interface, enter the **ipv6 access-group** command, specifying the ACL that you are applying to the interface, and **in**.

   ```
   device(config-Management-0k)# ipv6 access-group stdV6ACL1 in
   ```

Removing a Layer 3 ACL from an interface

To suspend ACL rules, you can remove the ACL containing those rules from the interface to which it was applied. After removal, you can also delete the ACL.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface** command, specifying the interface type and name.

   ```
   device(config)# interface ethernet 5/2
   ```

3. Enter the **no {ip | ipv6} access-group** command.

   ```
   device(conf-if-eth-5/2)# no ipv6 access-group ip_acl_1 in
   ```

Applying rACLs to devices

Use this procedure for applying IPv4 and IPv6 receive-path ACLs (rACLs) at global configuration level, using the **{ ip | ipv6 } receive access-group** command.

(IPv4 rACLs only) Note the destination parameters in the rules contained in the ACL that you are applying:

- To filter only unicast, routed route-processor traffic, in the rules contained in the ACL that you apply with this command, specify **any** for the destination parameter.
- To filter all traffic (switched, routed, unicast, multicast, router-processor, and data-plane), specify a destination IP address,

1. Enter **configure terminal** to change to global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the { **ip** | **ipv6** } **receive access-group** command, specifying the ACL that you are applying.

   ```
   device(config)# ip receive access-group ipv4-receive-acl-example
   ```

3. (For multiple rACLs) To specify processing order, enter the { **ip** | **ipv6** } **receive access-group** command with the **sequence** option.

   ```
   device(config)# ipv6 receive access-group test-racl-v6-2 seq 20
   ```

The following example creates an IPv4 ACL, defines rules needed for an rACL, and applies the ACL to the device.

```
device# configure terminal
device(config)# ip access-list extended ipv4-receive-acl-example
device(conf-ipacl-ext)# hard-drop tcp host 10.0.0.1 any count
device(conf-ipacl-ext)# hard-drop udp any host 20.0.0.1 count
device(conf-ipacl-ext)# permit tcp host 10.0.0.2 any eq telnet count
device(conf-ipacl-ext)# permit tcp host 10.0.0.2 any eq bgp count
device(conf-ipacl-ext)# exit
device(config)# ip receive access-group ipv4-receive-acl-example
```

The following example creates an IPv6 ACL, defines rules needed for an rACL, and applies the ACL to the device.

```
device# configure terminal
device(config)# ipv6 access-list extended ipv6-receive-acl-example
device(conf-ipacl-ext)# deny tcp host 10::1 any count
device(conf-ipacl-ext)# deny udp any host 20::1 count
device(conf-ipacl-ext)# permit tcp host 10::2 any eq telnet count
device(conf-ipacl-ext)# permit tcp host 10::2 any eq bgp count
device(conf-ipacl-ext)# exit
device(config)# ipv6 receive access-group ipv6-receive-acl-example
```

The following example creates two IPv4 extended ACLs, defines rules in the ACLs, and applies them as receive-path ACLs—specifying the priority of each ACL.

```
device#configure terminal
device(config)# ip access-list extended test-racl-1
device(conf-ipacl-ext)# deny ip 2.2.2.2/32 1.1.1.1/32
device(config)# ip access-list extended test-racl-2
device(conf-ipacl-ext)# permit ip 2.2.2.2/32 any
device(conf-ipacl-ext)# exit

device(config)#ip receive access-group test-racl-1 seq 10
device(config)#ip receive access-group test-racl-2 seq 20
```

Removing an rACL from a device

To suspend rACL rules, you can remove the ACL containing those rules from the device to which it was applied. After removal, you can also delete the ACL.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **no** { **ip** | **ipv6** } **receive access-group** command, specifying the ACL name.

   ```
   device(config)# no ip receive access-group ipv4-receive-acl-example
   ```

Layer 3 ACL modification

You can replace the contents of an ACL rule. You can also modify ACL sequence (**seq**) numbers.

Modifying Layer 3 ACL rules

To modify an ACL rule, delete the original rule and replace it with a new rule.

1. To display the rules of all ACLs of a given IP type and standard/extended specification, in global configuration mode enter the **show running-config** command.

```
device# show running-config ip access-list standard
ip access-list standard a1
 seq 10 permit host 10.1.1.1 count
```

Note the **seq** number of the rule that you need to delete or modify.

2. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

3. Enter the **{ip | ipv6} access-list** command, specifying the ACL you need to modify.

```
device(config)# ip access-list standard a1
```

4. Delete the original rule, doing one of the following:

- Enter the **no seq** command, specifying the sequence number of the rule that you are deleting.
```
device(conf-ipacl-std)# no seq 10
```

- Enter the exact rule that you are deleting, preceded by **no**.
```
no permit host 10.1.1.1 count
```

5. Enter the replacement rule.

```
device(conf-ipacl-std)# seq 10 permit host 10.1.1.1 log
```

Reordering the sequence numbers in a Layer 3 ACL

Reordering ACL-rule sequence numbers is helpful if you need to insert new rules into an ACL in which there are not enough available sequence numbers.

> **Note**
> Although you can use this procedure for IPv4 or IPv6 ACLs, the example is for IPv4.

Note the following regarding sequence numbers and their reordering parameters:

- The default initial sequence number is 10 and the default increment is 10.
- For reordering the sequence numbers, you need to specify the following:
  ◦ The new starting sequence number
  ◦ The increment between sequence numbers

The first rule receives the number of the starting sequence number that you specify. Each subsequent rule receives a number larger than the preceding rule. The difference in numbers is determined by the increment number that you specify. The starting-sequence number can range from 1 through 65535; the increment can also range from 1 through 65534.

For example: In the command below, for the IPv4 ACL "a1", the **resequence access-list** command assigns a sequence number of 5 to the first rule, 10 to the second rule, 15 to the third rule, and so forth.

```
device# resequence access-list ip a1 5 5
```

Advanced Layer 3 ACL rules and features

Many advanced ACL features are implemented per ACL rule, according to parameters that you specify.

> **Note**
> Some advanced features also require global configuration.

The following table describes advanced rule keywords for all supported devices.

Table 16: Layer 3 ACL advanced keywords

Keyword	Description	IPv4 standard ACL	IPv6 standard ACL	IPv4 extended ACL	IPv6 extended ACL	Comments
`copy-sflow`	sFlow monitoring	P/D; I	P/D; I	P/D; I	P/D; I	
`count`	Counter statistics	P/D/H; I/O	P/D/H; I	P/D/H; I/O	P/D/H; I	
`drop-precedence-force`	Re-marking drop-precedence	NA	NA	P; I	P; I	Only under **default**, **vxlan-visibility**, and **border-routing** TCAM profiles.
`dscp`	DSCP filtering	NA	NA	P/D/H; I/O	P/D/H; I	
`dscp-force`	DSCP re-marking	NA	NA	P; I	P; I	For routed traffic only.
`log` (SLX 9150, SLX 9250)	Logging	P/D/H; I	P/D/H; I	P/D/H; I	P/D/H; I	
`log` (SLX 9540, SLX 9640)	Logging	P/D; I	P/D; I	P/D; I	P/D; I	

Table 16: Layer 3 ACL advanced keywords (continued)

Keyword	Description	IPv4 standard ACL	IPv6 standard ACL	IPv4 extended ACL	IPv6 extended ACL	Comments
mirror (SLX 9150, SLX 9250)	Mirroring	NA	NA	P/D/H; I	P/D/H; I	Not supported for: • rACLs (receive-path) • ACL-RL (rate-limiting)
mirror (SLX 9540, SLX 9640)	Mirroring	NA	NA	P/D; I	P/D; I	Not supported for: • PBR ACLs (policy-based routing) • rACLs (receive-path) • ACL-RL (rate-limiting)

Key:

- **P**—Supported in a permit rule
- **D**—Supported in a deny rule
- **H**—Supported in a hard-drop rule
- **I**—Supported in an ACL applied to incoming traffic
- **O**—Supported in an ACL applied to outgoing traffic
- **NA**—Not available

For details, refer to the following *Extreme SLX-OS Command Reference* topics:

- seq (rules in IPv4 standard ACLs)
- seq (rules in IPv4 extended ACLs)
- seq (rules in IPv6 standard ACLs)
- seq (rules in IPv6 extended ACLs)

Parsing priorities among keywords

There are parsing priorities among the **copy-sflow**, **log**, and **mirror** keywords, as follows:

- Although in a standard-ACL rule you can include **log** and **copy-sflow**, only one of the two is processed, as follows:
  - In a permit rule, the order of precedence is **copy-sflow** > **log**.
  - In a deny or hard-drop rule, the order of precedence is **log** > **copy-sflow**.
- Although in an extended-ACL rule you can include **log**, **mirror**, and **copy-sflow**, only one of the three is processed, as follows:
  - In a permit rule, the order of precedence is **mirror** > **copy-sflow** > **log**.
  - In a deny or hard-drop rule, the order of precedence is **log** > **copy-sflow** > **mirror**.

Consider the following extended IPv4 ACL:

```
device(config)# ip access-list extended ip_acl_01
device(conf-ipacl-ext)# seq 10 permit host 10.24.26.145 any count log mirror copy-sflow
device(conf-ipacl-ext)# seq 20 deny host 10.34.36.245 any count log mirror copy-sflow
```

- In the permit rule, only the **mirror** keyword is processed.
- In the deny rule, only the **log** keyword is processed.

Filter and Force DSCP Values (IPv4 ACLs)

In IPv4 extended ACL rules, re-marking (forcing) DSCP values can change the priority on egress traffic, by which you can prioritize ingress traffic.

You can also filter IPv4 packets by DSCP value.

1. Access global configuration mode.

```
device# configure
```

2. Create or access the ACL.

```
device(config)# ip access-list extended extd_ACL5
```

3. To filter incoming or outgoing packets by DSCP value, define **permit** or **deny** rules specifying the **dscp** parameters.

```
device(config-ipacl-ext)# seq 5 deny tcp host 10.24.26.145 any dscp 25
device(config-ipacl-ext)# seq 15 permit tcp 10.24.26.146 any dscp 20
```

4. To re-mark the DSCP value of incoming packets, define **permit** rules specifying the **dscp-force** parameters.

```
device(config-ipacl-ext)# seq 25 permit tcp 10.24.26.147 any dscp-force 10
```

> **Note**
> (SLX 9740 only) If traffic is bridged, then the egressing PCP (802.1p) is re-marked to the equivalent value of the forced DSCP.

5. Apply the ACL to the appropriate interface.

```
device(config)# interface ethernet 2/2
device(conf-if-eth-2/2)# ip access-group extd_ACL5 in
```

Filtering and forcing DSCP values (IPv6 ACLs)

In IPv6 extended ACL rules, re-marking (forcing) DSCP values can change priority on egress traffic, by which you can prioritize ingress traffic. You can also filter IPv6 packets by DSCP value.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ipv6 access-list extended** command to create or access the ACL.

```
device(config)# ipv6 access-list extended ipv6_acl_1
```

3. To filter incoming packets by DSCP value, define **permit** or **deny** rules specifying the **dscp** parameters.

```
device(conf-ip6acl-ext)# seq 10 deny ipv6 2001:2002:1234:1::/64 2001:1001:1234:1::/64
dscp 25 count
device(conf-ip6acl-ext)# seq 20 permit ipv6 2001:2002:2345:1::/64 any dscp 20 count
```

4. To re-mark the DSCP value of incoming packets, define **permit** rules specifying the **dscp-force** parameters.

```
device(conf-ip6acl-ext)# seq 30 permit ipv6 2001:2002:2346:1::/64 any dscp-force 10
```

5. Apply the ACL that you created to the appropriate interface.

```
device(config)# interface ethernet 2/2
device(conf-if-eth-2/2)# ipv6 access-group ipv6_acl_1 in
```

ACL logs

ACL logs can provide insight into permitted and denied network traffic.

ACL logs maintain the following properties:

- Supported for all ACL types (MAC, IPv4, and IPv6)
- Supported for incoming network traffic only
- Supported for all user interfaces (but not on management interfaces) on which ACLs can be applied
- May be CPU-intensive

You can also enable Raslog logging for ACLs.

Enabling and configuring the ACL log buffer

Among the conditions required for ACL logging is that the ACL log buffer be enabled and configured.

1. Enter the **debug access-list-log buffer** command to enable and configure ACL log buffering.

```
device# debug access-list-log buffer circular packet count 1600
```

2. (Optional) To display the current ACL log buffer configuration, enter the **show access-list-log buffer config** command.

```
device# show access-list-log buffer config
ACL Logging is enabled
Buffer exists for interface Eth 1/11
Buffer type is Circular and size is 1000
```

Enabling IPv4 ACL rules for logging

When you create ACL rules for which you want to enable logging, you must include the **log** parameter.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ip access-list** command to create or modify an access list.

```
device(config)# ip access-list standard ip_acl_1
```

3. For each ACL rule for which you need logging, include the **log** keyword.

```
device(conf-ipacl-std)# seq 5 permit host 10.20.33.4 log
```

4. Apply the ACL that you created to the appropriate interface.

5. (Optional) To display ACL logs, enter the **show access-list log buffer** command.

```
device# show access-list-log buffer
Frames Logged on interface 2/1 :
-------------------------------
Frame Received Time : Fri Dec 9 3:8:48 2011
Ethernet,       Src : (00:34:56:78:0a:ab), Dst: (00:12:ab:54:67:da)
```

```
  Ethtype              : 0x8100
  Vlan tag type        : 0x800
  VlanID               : 0x1
Internet proto, Src : 192.85.1.2, Dst: 192.0.0.1
  Interface            :
  Type of service      : 0
  Length               : 110
  Identification       : 0
  Fragmentation        : 00 00
  TTL                  : 255
  protocol             : 253
  Checksum             : 39 3a
  Payload type         :
packet(s) repeated  : 30
Ingress Deny Logged
```

> **Note**
> If an ACL with rules that contain the **log** keyword is applied to the management interface, logs are not recorded for that ACL.

Enabling IPv6 ACL rules for logging

When you create ACL rules for which you want to enable logging, you must include the **log** parameter.

1. Enter the **configure** command to access global configuration mode.

   ```
   device# configure
   ```

2. Enter the **ipv6 access-list** command to create or modify an access list.

   ```
   device(config)# ipv6 access-list extended ipv6_acl_1
   ```

3. For each ACL rule for which you need logging, include the **log** keyword.

   ```
   device(conf-ip6acl-ext)# seq 20 deny ipv6 2002:2003:1234:1::/64 2001:3001:1234:1::/64
   log
   ```

4. Apply the ACL that you created to the appropriate interface.

5. (Optional) To display ACL logs, enter the **show access-list log buffer** command.

   ```
   device# show access-list-log buffer
   Frames Logged on interface Eth 2/1 :
   -----------------------------------
   Frame Received Time  : Wed Apr 6 2016 8:15:4
   Ethernet,     SrcMAC : 00:24:38:9b:cf:21, DstMAC: 76:8e:f8:05:70:14
     Ethtype            : 0x86dd

   Protocol Type        : IPV6
   SrcIP                : 26::1
   DstIP                : 25::1
   Interface            : Eth 1/16
   Flow-ID              : 63800000
   Payload Length       : 1c6
   Nxt Header Type      : 6 (TCP)
   Hop-Limit            : 63

   packet(s) repeated   : 11565
   ```

```
Ingress Deny Logged
----------------------------------------------------------------------------
```

> **Note**
> If an ACL with rules that contain the **log** keyword is applied to the management interface,
> logs are not recorded for that ACL.

Enabling and configuring ACL Raslogs

This task enables Raslog messages for ACL rules with **log** keywords and specifies how long the system
waits between ACL Raslog messages.

For details of Raslog messages, refer to the *Extreme SLX-OS Message Reference*.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **acl-policy** command to access ACL policy configuration mode.

   ```
   device(config)# acl-policy
   ```

3. Enter the **acl-log-raslog** command to enable ACL Raslogs.

   ```
   device(config-acl-policy)# acl-log-raslog
   ```

4. To modify the interval between the first ACL Raslog and each consecutive Raslog, enter the **acl-
   log-raslog log-interval** command.

   ```
   device(config-acl-policy)# acl-log-raslog log-interval 8
   ```

The following output is an ACL Raslog example.

```
MAC ACL mac_2 permitted 1 packets on intf eth1/6 [SA:0010.1010.1001, DA:0001.0300.0500,
Type:0, VLAN:101, SIP:0.0.0.0, DIP:0.0.0.0, l3_proto:none, src_port:0, dst_port:0]

IP ACL v4acl denied 1 packets on intf eth1/6  [SA:0001.0300.0400,DA:0001.0300.0500,
Type:800, VLAN:100, SIP:2.2.2.2, DIP:6.6.6.6, l3_proto:udp, src_port:66, dst_port:77]

IPv6 ACL v6acl permitted 1 packets on intf po44  [SA:0001.0300.0400,DA:0001.0300.0500,
Type:86dd, VLAN:100, SIP:fe80::201:3ff:fe00:400,
DIP:3555:5555:6666:6666:7777:7777:8888:8888,
l3_proto:udp, src_port:63, dst_port:63]
```

Layer 3 ACL-based mirroring

ACL-based mirroring enables you to monitor specified inbound traffic in the mirrored port by attaching
a protocol analyzer to it.

ACL mirroring is supported for *ethernet*, *port-channels*, and *ve* ingress interfaces.

Enabling IPv4 ACL rules for mirroring

ACL-based inbound mirroring applies to extended-ACL rules that include the **mirror** keyword.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **ip access-list extended** command to create or access the ACL.

   ```
   device(config)# ip access-list extended extd_ACL5
   ```

3. In each rule for which you need to enable mirroring, include the **mirror** keyword.

```
device(conf-ipacl-ext)# seq 5 deny tcp host 10.24.26.145 any mirror
device(conf-ipacl-ext)# seq 15 permit tcp 10.24.26.146 any mirror
```

4. Apply the ACL that you created to the appropriate physical interface, specifying the **in** keyword.

```
device(config)# interface ethernet 2/1
device(conf-if-eth-2/1)# ip access-group extd_ACL5 in
```

Enabling IPv6 ACL rules for mirroring

ACL-based inbound mirroring applies to extended-ACL rules that include the **mirror** keyword.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ipv6 access-list extended** command to create or access the ACL.

```
device(config)# ipv6 access-list extended extd_v6_01
```

3. In each rule for which you need to enable mirroring, include the **mirror** keyword.

```
device(conf-ipv6acl-ext)# seq 5 permit tcp 1000:1::/64 2000:1::/64 mirror
device(conf-ipacl-ext)# seq 15 deny udp 1000:1::/64 2000:1::/64 mirror
```

4. Apply the ACL that you created to the appropriate physical interface, specifying the **in** keyword.

```
device(config)# interface ethernet 3/1
device(conf-if-eth-3/1)# ipv6 access-group extd_v6_01 in
```

Defining an ACL mirror port

To support ACL mirroring, define a destination ACL mirror port common to all ports of a port processor (PPCR). ACL mirroring is supported for *ethernet*, *port-channels*, and *ve* ingress interfaces.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. To define a physical interface as the mirror, enter a command such as the following example.

```
device(config)# acl-mirror source ethernet 0/1 destination ethernet 0/2
```

3. To define a port-channel as the mirror, enter a command such as the following example.

```
device(config)# acl-mirror source ethernet 0/1 destination port-channel 1
```

4. To define a port-channel as the source for the mirror, enter a command such as the following example.

```
device(config)# acl-mirror source port-channel 3 destination port-channel 6
```

ACL counter statistics (Layer 3)

If an ACL rule contains the **count** parameter, you can access statistics for the rule, including the number of frames permitted or denied by that rule. If needed, you can also clear ACL statistics.

Creating an IPv4 ACL rule enabled for counter statistics

When you create ACL rules, the **count** parameter enables you to display counter statistics.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ip access-list** command to create or modify an access list.

```
device(config)# ip access-list standard stdACL3
```

3. For each ACL rule for which you need to display statistics, include the **count** keyword.

```
device(conf-ipacl-std)# seq 5 permit host 10.20.33.4 count
device(config-ipacl-std)# seq 15 deny any count
```

4. If you have not yet applied the ACL to the appropriate interface, do so now.

Creating an IPv6 ACL rule enabled for counter statistics

When you create ACL rules, the **count** parameter enables you to display counter statistics.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ipv6 access-list** command to create or modify an access list.

```
device(config)# ipv6 access-list extended ip_acl_1
```

3. For each ACL rule for which you need to display statistics, include the **count** keyword.

```
device(conf-ip6acl-ext)# seq 20 deny ipv6 2002:2003:1234:1::/64 2001:3001:1234:1::/64
count
```

4. If you have not yet applied the ACL to the appropriate interface, do so now.

5. (Optional) To display ACL counter statistics, enter the **show statistics access-list** command.

```
device# show statistics access-list ipv6 ip_acl_1 in
ipv6 access-list ip_acl_1 on Ethernet 2/3 at Ingress (From User)
    seq 10 deny ipv6 2001:2002:1234:1::/64 2001:1001:1234:1::/64 count (0 frames)
    seq 20 deny ipv6 2002:2003:1234:1::/64 2001:3001:1234:1::/64 count (33 frames)
```

The following example shows how to create an IPv6 extended ACL and define a counter-enabled rule for it.

```
device# configure terminal
device(config)# ipv6 access-list extended ip_acl_1
device(conf-ip6acl-ext)# seq 10 deny ipv6 2001:2002:1234:1::/64 2001:1001:1234:1::/64
count
```

Enabling IPv4 ACL rules for sFlow monitoring

sFlow is a sampling technology for monitoring networks. You can monitor specified incoming data flows by including the **copy-sflow** keyword in rules within an ACL applied to a device.

In order for sFlow to function, the sFlow collector must be globally configured. For details, refer to the *Extreme SLX-OS Monitoring Configuration Guide*.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ip access-list standard/extended** command to create or access the ACL.

```
device(config)# ip access-list extended ext-vfour1
```

3. In each rule for which you need to enable sFlow, include the **copy-sflow** keyword.

```
device(conf-ipacl-ext)# permit 30.30.30.0 255.255.255.0 any copy-sflow
device(conf-ipacl-ext)# deny 31.31.31.0 255.255.255.0 copy-sflow
```

4. Apply the ACL that you created to the appropriate physical interface, specifying the **in** keyword.

```
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ip access-group ext-vfour1 in
```

Enabling IPv6 ACL rules for sFlow monitoring

sFlow is a sampling technology for monitoring networks. You can monitor specified incoming data flows by including the **copy-sflow** keyword in rules within an ACL applied to a device.

In order for sFlow to function, the sFlow collector must be globally configured. For details, refer to the *Extreme SLX-OS Monitoring Configuration Guide*.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ipv6 access-list standard/extended** command to create or access the ACL.

```
device(config)# ipv6 access-list extended ipv6_ac1_1
```

3. In each rule for which you need to enable sFlow, include the **copy-sflow** keyword.

```
device(conf-ip6acl-ext)# seq 20 deny ipv6 2002:2003:1234:1::/64 2001:3001:1234:1::/64
copy-sflow
device(conf-ip6acl-ext)# seq 30 permit ipv6 2002:2004:1234:1::/64
2001:3002:1234:1::/64 copy-sflow
```

4. Apply the ACL that you created to the appropriate physical interface, specifying the **in** keyword.

```
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ipv6 access-group ipv6_ac1_1 in
```

Disabling conflicting-rule check

Towards editing ACLs, you can disable the default restriction on conflicting rules within an ACL. You can then create a conflicting rule before deleting the previous version.

> **Note**
> We recommend that after ACL-editing sessions towards which you disabled conflicting-rule check, restore the default setting—by entering the **no allow-conflicting-rules** command.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter **acl-policy** to access ACL policy mode.

```
device(config)# acl-policy
```

3. Enter **allow-conflicting-rules**.

```
device(config-acl-policy)# allow-conflicting-rules
```

The following example is a typical editing flow.

1. Enter the **show running-config** command to display the rules in the ACL that you need to modify.

```
device# show running-config mac access-list extended mac1
mac access-list extended mac1
 seq 10 permit host 0001.0001.0001 any
```

```
  seq 20 deny host 0001.0001.0002 any count
  seq 30 hard-drop host 0001.0001.0003 any mirror
```

2. Enter the **allow-conflicting-rules** command.

```
device# configure terminal
device(config)# acl-policy
device(config-acl-policy)# allow-conflicting-rules
```

3. In the ACL that you need to modify, create the new rule and then delete the old rule.

```
device(config-acl-policy)# exit
device(config)# mac access-list mac1
device(conf-macl-ext)# seq 21 permit host 0001.0001.0002 any count
device(conf-macl-ext)# no seq 20
```

4. Enter the **no allow-conflicting-rules** command to restore the default setting.

```
device(conf-macl-ext)# exit
device(config)# acl-policy
device(config-acl-policy)# no allow-conflicting-rules
```

Disabling duplicate-rule check

Towards editing ACLs, you can disable the default restriction on duplicate rules within an ACL. You can then create a duplicate rule at a new sequence before deleting the previous version.

Note

We recommend that after ACL-editing sessions towards which you disabled duplicate-rule check, restore the default setting—by entering the **no allow-duplicate-rules** command.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter **acl-policy** to access ACL policy mode.

```
device(config)# acl-policy
```

3. Enter **allow-duplicate-rules**.

```
device(config-acl-policy)# allow-duplicate-rules
```

The following example is a typical editing flow.

1. Enter the **show running-config** command to display the rules in the ACL that you need to modify.

```
device# show running-config mac access-list extended mac1
mac access-list extended mac1
  seq 10 permit host 0001.0001.0001 any
  seq 20 deny host 0001.0001.0002 any count
  seq 30 hard-drop host 0001.0001.0003 any mirror
```

2. Enter the **allow-duplicate-rules** command.

```
device# configure terminal
device(config)# acl-policy
device(config-acl-policy)# allow-duplicate-rules
```

3. In the ACL that you need to modify, create the duplicate rule—specifying the new sequence number —and then delete the old rule.

```
device(config-acl-policy)# exit
device(config)# mac access-list mac1
```

```
device(conf-macl-ext)# seq 11 hard-drop host 0001.0001.0003 any mirror
device(conf-macl-ext)# no seq 30
```

4. Enter the **no allow-duplicate-rules** command to restore the default setting.

```
device(conf-macl-ext)# exit
device(config)# acl-policy
device(config-acl-policy)# no allow-duplicate-rules
```

ACL show and clear commands

There is a full range of ACL show and clear commands, listed here with descriptions.

Table 17: ACL show commands in the *Command Reference*

Command	Description
`show access-list`	For a given network protocol and inbound/outbound direction, displays ACL information. You can show information for a specified ACL or only for that ACL on a specified interface. You can also display information for all ACLs bound to a specified interface.
`show access-list-log buffer`	Displays the contents of the ACL log buffer.
`show access-list-log buffer config`	Displays the ACL log buffer configuration.
`show running-config {mac	ip	ipv6} access-list`	For a given network protocol and standard/extended type, displays ACL configuration. You can show the configuration of a specified ACL or for all such ACLs.
`show statistics access-list`	For a given network protocol and inbound/outbound direction, displays statistical information—for ACL rules that include the **count** keyword. You can show statistics for a specified ACL or only for that ACL on a specified interface. You can also display statistical information for all ACLs bound to a specified interface or to the management interface.

Table 18: ACL clear commands in the *Command Reference*

Command	Description
`clear counters access-list`	For a given network protocol and inbound/outbound direction, clears ACL statistical information. You can clear all statistics for a specified ACL or only for that ACL on a specified interface. You can also clear statistical information for all ACLs bound to a specified interface or to the management interface.

IP broadcast ACLs (bACLs)

IP broadcast ACLs (bACLs) provide hardware-based filtering of IP subnet-based directed broadcast and network-address traffic.

> **Note**
> Broadcast ACLs are not supported on SLX 9150, or SLX 9250 devices.

bACLs identify directed broadcast and network-address traffic by the specified subnets, and filter traffic on the corresponding VRF. The bACL implementation flow is as follows:

1. Create a standard or extended IPv4 ACL.
2. Within the ACL, define needed permit/deny rules.
3. Apply the ACL at device level, interface level, or VE level.

This flow programs ACL entries in the CAM for each configured broadcast address and network address, eliminating the need to define rules for each trusted source/destination subnet combination.

Configuration guidelines for bACLs

The configuration considerations for bACLs are as follows:

- You can apply no more than one bACL at device level.
- You can apply no more than one bACL to an interface or VE.
- bACLs applied at device level filter traffic both on default-VRF and user-vrf interfaces.
- If a physical port is a member of a virtual interface, then ACL binding is permitted only at the VE level and not at the physical port level.
- If bACLs are applied both at device level and at interface- or VE-level, a match at interface- or VE-level takes precedence over a match at device level.
- For LAG ports, all ports within the LAG must have the same IP broadcast ACL applied to them before the LAG is created. On deleting the LAG, the IP broadcast ACL binding is replicated on all individual LAG ports.
- IP directed-broadcast ACL binding is not be permitted on VPLS and VLL endpoints.
- IP tunnel interfaces are not supported.
- Because ingress traffic matching bACLs is not subject to security ACLs or RL-ACLs, configure bACLs so that only directed broadcast traffic matches the bACL permit/deny/hard-drop rules.
- bACLs do not support ACL-based logging, Sample Flow (sFlow), or mirroring.
- Traffic matching bACLs is not subject to policy-based routing (PBR).
- CAM sharing is not supported for bACLs.
- Broadcast ACLs do not filter /31 addresses—because the /31-subnet supports only two unicast addresses.
- When ACLs of multiple types are applied, processing priority is as follows: bACLs > rACLs > PBR > Layer 3 ACLs > Layer 2 ACLs. However, if any filter has a drop match, the packet is dropped irrespective of the priority.

Creating a standard bACL

A standard ACL permits or denies traffic according to source address only.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ip access-list standard** command to create the ACL.

```
device(config)# ip access-list standard stdACL3
```

3. Enter rules crafted for your IP broadcast ACL (bACL).

```
device(conf-ipacl-std)# permit host 10.1.1.2
```

Creating an extended bACL

An extended ACL permits or denies traffic according to one or more parameters, including source address, destination address, port, protocol (TCP or UDP), and TCP flags.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ip access-list extended** command to create the ACL.

```
device(config)# ip access-list extended extdACL5
```

3. Enter rules crafted for your IP broadcast ACL (bACL).

```
device(conf-ipacl-ext)# permit ip 2.2.2.2/32 any
```

Applying a bACL to a device

Use this procedure for applying an IP broadcast ACL (bACL) at global configuration level.

1. Enter **configure terminal** to change to global configuration mode.

```
device# configure terminal
```

2. Enter the **ip global-subnet-broadcast-acl** command, specifying the bACL that you are applying.

```
device(config)# ip global-subnet-broadcast-acl bacl_10
```

Applying a bACL to a physical interface

Use this procedure for applying an IP broadcast ACL (bACL) to a physical interface.

1. Enter **configure terminal** to change to global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command, specifying the port number.

```
device(config)# interface ethernet 0/2
```

3. Enter the **ip subnet-broadcast-acl** command, specifying the bACL that you are applying to the interface.

```
device(conf-if-eth-0/2)# ip subnet-broadcast-acl bacl_10
```

Applying a bACL to a VE interface

Use this procedure to apply an IP broadcast ACL (bACL) to a VE interface (attached to a VLAN).

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface ve** command, specifying the *vlan-id*.

   ```
   device(config)# interface ve 50
   ```

3. Enter the **ip subnet-broadcast-acl** command, specifying the bACL that you are applying to the VE.

   ```
   device(config-ve-50)# ip subnet-broadcast-acl bacl_20
   ```

bACL configuration example

The following diagram illustrates how filtering of IP directed broadcast traffic is enabled on the Router 3 interface. To enable filtering of IP directed broadcast traffic on Router 3 interface 1/2, configure a bACL permitting source IP address 10.1.1.2, and apply it that interface. Router 3 then allows IP broadcast packets from 10.1.1.2 and drops IP broadcast packets from other sources.
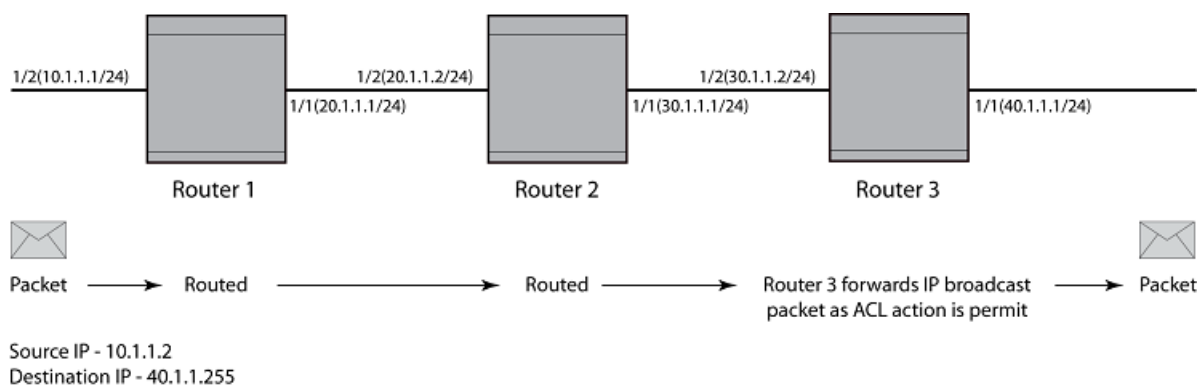
Figure 4: Filtering of IP directed broadcast traffic on the Router 3 interface

To configure a bACL on Router 3 interface 1/2, enter the following commands.

```
device# configure terminal
device(config)# ip access-list standard bACL_int_3
device(conf-ipacl-std)# permit host 10.1.1.2
device(conf-ipacl-std)# exit
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# enable
device(conf-if-eth-1/1)# ip address 10.1.1.1/24
device(conf-if-eth-1/1)# exit
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# enable
device(conf-if-eth-1/2)# ip address 10.1.1.1/24
device(conf-if-eth-1/2)# ip subnet-broadcast-acl bACL_int_3
device(conf-if-eth-1/2)# exit
```

bACL show and clear commands

There are show and clear commands supporting IP broadcast ACLs (bACLs), listed here with descriptions.

Table 19: bACL show commands in the *Command Reference*

Command	Description
`show access-list subnet-broadcast ip`	Displays information for bACLs applied to interfaces. You can show information for all such bACLs, for a specific bACL, or for that bACL on a specific physical or VE interface.
`show access-list global-subnet-broadcast`	Displays information for a specific bACL applied at device level.
`show statistics access-list subnet-broadcast`	Displays statistical information for bACLs applied to interfaces. You can show information for all such bACLs, for a specific bACL, or for that bACL on a specific physical or VE interface.
`show statistics access-list global subnet-broadcast`	Displays statistical information for a specific bACL applied at device level.

Table 20: ACL clear commands in the *Command Reference*

Command	Description
`clear counters access-list subnet-broadcast`	Clears bACL statistical information for bACLs applied to interfaces. You can clear statistics for all such bACLs, for a specific ACL, or only for that bACL on a specific physical or VE interface.
`clear counters global-subnet-broadcast`	Clears bACL statistical information for a specific bACL applied at device level.

Policy-Based Routing

Policy-Based Routing Overview on page 76
Route maps on page 77
Configuring a PBR policy on page 78

Policy-Based Routing Overview

Policy-Based Routing (PBR) is the process of altering a packet's path based on criteria other than the destination address. PBR allows you to use ACLs and route maps to selectively route an IP packet. The ACLs classify the traffic and the route maps that match on the ACLs set routing attributes for the traffic.

A PBR policy specifies the routing attributes for traffic that matches the policy. Using standard ACLs with route maps in PBR, an IP packet is routed based on their source IP address. With extended ACLs, you can route IP packets based on all of the clauses in the extended ACL.

> **Note**
> For more details about ACLs, refer to the "ACLs" section.

You can configure the device to perform the following types of PBR based on a packet Layer 3 and Layer 4 information:

- Select the next-hop gateway.
- Send the packet to the null interface (null0).

To configure PBR, you define the policies using ACLs and route maps, then enable PBR on individual interfaces. The platform programs the ACLs on the interfaces, and routes traffic that matches the ACLs according to the instructions provided by the "set" statements in the route map entry.

The configuration of a set of match criteria and corresponding routing information (for example next hops) is referred to as a stanza. You can create multiple stanzas and assign an "Instance_ID" that controls the program positioning within the route map. Furthermore, when the route map is created, you specify a deny or permit construct. In addition, the ACL used for the "match" criteria also contains a deny or permit construct.

The deny or permit nomenclature has a different meaning within the context of the PBR operation than it does within the normal context of security ACLs (where deny and permit are directly correlated to the forwarding actions of forward and drop). The following table lists the behavior between the permit and

deny actions specified at the route-map level, in conjunction with the permit and deny actions specified at the ACL rule level.

Table 21: Behavior of Permit and Deny actions in different contexts

Route-map level permit and deny actions	ACL clause permit and deny actions	Resulting Ternary Content Addressable Memory (TCAM) action
Permit	Permit	The "set" statement of the route-map entry is applied.
Permit	Deny	The packet is "passed" and routed normally. The contents of the "set" command are not applied. A rule is programmed in the TCAM as a "permit" with no result actions preventing any further statements of the route-map ACL from being applied.
Deny	Permit	The packet is "passed" and routed normally. There should be no "set" commands following the "match" command of a deny route-map. A rule is programmed in the TCAM as a "permit" with no result actions preventing any further statements of the route-map ACL from being applied.
Deny	Deny	No TCAM entry is provisioned; no other route-map ACL entries will be compared against. If no subsequent matches are made, the packet is forwarded as normal.

Route maps

Route maps enable you to define routing policy for the traffic causing a packet to be forwarded to a predetermined next-hop interface, bypassing the path determined by normal routing.

Each entry in a route map statement contains a combination of match and set statements. A route map specifies the match criteria that correspond to ACLs, followed by a set statement that specifies the resulting action if all of the match clauses are met. You can define multiple match and next-hop specifications (set statement) on the same interface. When a PBR policy has multiple next-hops to a destination, PBR selects the first live next-hop specified in the policy that is up. If none of the policy's direct routes or next-hops is available, the packets are forwarded as per the routing table.

Match statement

You can use standard or extended ACLs to establish the match criteria. Using standard ACLs with route maps in PBR, an IP packet is routed based on their source IP address. Using extended ACLs, you can route IP packets based on all of the clauses in the extended ACL.

Set statement

Traffic that matches a match statement in the route map is forwarded as defined by set commands. Multiple set commands can be configured and when a match condition is met, the device works sequentially through the list of set commands until it finds the first next-hop that is operational and uses it. If that next-hop goes down, the next-hop as defined in a set command is chosen and if all next-hop interfaces in the list are down, the packet is routed as determined in the IP Route Table. If a next-

hop interface that was down comes back up, the next-hop selection process begins again and restarts its selection process from the top of the list.

The set clauses are evaluated in the following order:

1. Set clauses where the next-hop is specified.
2. Set interface NULL0.

The order in which you enter the **set ip next-hop** commands determines the order preference. If no next-hops are reachable, the egress interface is selected based on the order of interface configuration. The set interface NULL0 clause — regardless of which position it was entered — is always placed as the last selection in the list.

> **Note**
> The "match" and "set" statements described in this chapter are the only route-map statements supported for PBR. Other route-map statements described in the documentation apply only to the protocols with which they are described.

> **Note**
> If none of the clauses of a PBR routemap definition contains both 'match' and 'set' statements together, PBR does not work and the packets are forwarded as per the routing table.

The following are the PBR next-hops that can be specified in a route map for a matched traffic:

- IPv4 address
- IPv6 address
- Null interface

Configuring a PBR policy

To configure a PBR, you must define the policies using ACLs and route map, then enable PBR globally or on individual interfaces. The device programs the ACLs into the session CAM on the interfaces and routes traffic that matches the ACLs according to the instructions in the route maps.

The following are the basic steps to configure a PBR policy:

1. Configure ACLs that specify all the conditions required to match the desired packets to be routed using PBR.

   For details about configuring ACLs, see the "ACLs" section of this publication.
2. Configure a route map that matches on the ACLs and sets the route information.
3. Enable PBR by applying the route map on an interface.

Policy-based routing (IPv4)

This section provides configuration details of PBR policy that specifies to match and route IPv4 packets.

Configuration considerations and guidelines for PBR

- PBR route maps may only be applied to Layer 3 (L3) interfaces. Application of a route map to a non-L3 interface results in the configuration being rejected.
- PBR is not supported on any interface where the next-hop is configured to be a GRE Tunnel.
- Deletion of a route map or deletion of an ACL used in the route map "match" is not allowed when the route map is actively bound to an interface. Attempts to delete an active route map or associated ACL is rejected, and an error and log are generated.
- The "set" commands are only available within the context of a "permit" stanza.
- Consider the permit and deny keywords as allowing the specified match content as either being permitted to or denied from using the defined "set criteria" of the route map. The permit and deny keywords do not correlate to the forwarding action of forward and drop as they do in the ACL application.

Configuring an IPv4 PBR with IPv4 address as the next hop

The following steps configure an IPv4 PBR by setting an IPv4 address as the next hop in the route map.

> **Note**
> This task uses access-control lists (ACLs). For details about configuring ACLs, see the "ACLs" section of this publication.

1. Enter the **configure terminal** command to enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Define the required IPv4 ACLs to be added to the route map.

   ```
   device(config)# ip access-list standard 99
   device(conf-ipacl-std)# permit 192.168.0.0 255.255.255.0
   ```

3. Enter the **exit** command to return to global configuration mode.

   ```
   device(conf-ipacl-std)# exit
   ```

4. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all the match clauses are met.

   ```
   device(config)# route-map test-route permit 99
   ```

5. Add IPv4 ACLs to match the IP address that is permitted by the ACL.

   ```
   device(config-route-map-test-route/permit/99)# match ip address acl 99
   ```

6. Set the IPv4 address of the next hop to which the traffic that matches a match statement in the route map must be routed.

   ```
   device(config-route-map-test-route/permit/99)# set ip next-hop 192.168.3.1
   ```

   Optionally, you can specify the configured next hop address to be resolved from either the global routing table or VRF routing table using the **global** or **VRF** *vrf-name* options.

7. Enter the **exit** command to return to the global configuration mode.

   ```
   device(config-route-map-test-route/permit/99)# exit
   ```

8. Enable PBR by applying the route map on an interface or on virtual interface.

- Enable IPv4 PBR by applying the route map on an interface.

```
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ip policy route-map test-route
```

- Enable IPv4 PBR by applying the route map on a virtual interface.

```
device(config)# interface Ve 1
device(config-if-Ve-1)# ip policy route-map test-route
```

The following example shows the configuration steps to configure an IPv4 PBR by setting an IPv4 address as the next hop in the route map.

```
device# configure terminal
device(config)# ip access-list standard 99
device(conf-ipacl-std)# permit 192.168.0.0 255.255.255.0
device(conf-ipacl-std)# exit
device(config)# route-map test-route permit 99
device(config-route-map-test-route/permit/99)# match ip address acl 99
device(config-route-map-test-route/permit/99)# set ip next-hop 192.168.3.1
device(config-route-map-test-route/permit/99)# exit
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ip policy route-map test-route
device(conf-if-eth-1/1)# end
```

Configuring an IPv4 PBR with NULL0 interface as the next hop

The following steps configure an IPv4 PBR by setting NULL0 interface as the next hop in the route map.

1. Enter the **configure terminal** command to enter global configuration mode.

```
device# configure terminal
```

2. Define the required IPv4 ACLs to be added to the route map.

```
device(config)# ip access-list standard 99
device(conf-ipacl-std)# permit 192.168.0.0 255.255.255.0
```

3. Enter the **exit** command to return to global configuration mode.

```
device(conf-ipacl-std)# exit
```

4. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all the match clauses are met.

```
device(config)# route-map test-route permit 99
```

5. Add IPv4 ACLs to match the IP address that is permitted by the ACL.

```
device(config-route-map-test-route/permit/99)# match ip address acl 99
```

6. Set the next hop as NULL0 interface to send the traffic to the null interface, thus dropping the packet instead of forwarding it.

```
device(config-route-map-test-route/permit/99)# set ip interface null0
```

7. Enter the **exit** command to return to the global configuration mode.

```
device(config-routemap-test-route/permit/99)# exit
```

8. Enter configuration mode on the interface where you want to enable PBR by applying the route map.

```
device(config)# interface ethernet 1/1
```

9. Enable policy-based routing on the interface and specify the route map to be used.

```
device(conf-if-eth-1/1)# ip policy route-map test-route
```

The following example shows the configuration steps to configure an IPv4 PBR to send all traffic from 10.157.23.0 0.0.0.255 to the null interface, thus dropping the traffic instead of forwarding it.

```
device# configure terminal
device(config)# evice(config)# ip access-list standard 99
device(conf-ipacl-std)# permit 10.157.23.0 0.0.0.255
device(conf-ipacl-std)# exit
device(config)# route-map test-route permit 99
device(config-route-map-test-route/permit/99)# match ip address acl 99
device(config-route-map-test-route/permit/99)# set ip interface null0
device(config-route-map-test-route/permit/99)# exit
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ip policy route-map test-route
device(conf-if-eth-1/1)# end
```

Configuration examples for policy based routing

This section presents configuration examples for configuring and applying a PBR policy.

Policy-Based Routing with differing next hops

In this example, traffic is routed from different sources to different places (next hops). Packets arriving from source 192.168.0.0 are sent to the VRF vroutefwd's next hop at 3.3.3.3; packets arriving from source 192.168.1.1 are sent to the VRF vroutefwd's next hop at 3.3.3.5.

1. Configure the ACLs.

```
device(config)# ip access-list standard Jules
device(conf-ipacl-std)# permit 192.168.0.0 255.255.255.0
device(conf-ipacl-std)# exit
device(config)# ip access-list standard Vincent
device(conf-ipacl-std)# permit 192.168.1.1 255.255.255.0
device(conf-ipacl-std)# exit
```

2. Create the first stanza of the route map. (The example is using a route-map named pulp_fiction.)

```
device(config)# route-map pulp_fiction permit 10
device(config-routemap-pulp_fiction/permit/10)# match ip address acl Jules
device(config-routemap-pulp_fiction/permit/10)# set ip vrf vroutefwd next-hop 3.3.3.3
device(config-routemap-pulp_fiction/permit/10)# exit
```

3. Create the second stanza of the route-map (The example is using a route-map named pulp_fiction.)

```
device(config)# route-map pulp_fiction permit 20
device(config-routemap-pulp_fiction/permit/20)# match ip address acl Vincent
device(config-routemap-pulp_fiction/permit/20)# set ip vrf vroutefwd next-hop 3.3.3.5
device(config-routemap-pulp_fiction/permit/20)# set ip next-hop 6.6.6.7
device(config-routemap-pulp_fiction/permit/20)# exit
```

4. Bind the route map to the desired interface.

```
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ip policy route-map pulp_fiction
```

Policy-Based Routing and NULL0 with match statements

NULL0 is a mechanism used to drop packets in the Policy-Based Routing (PBR). If the NULL0 interface is specified within a stanza and the stanza also contains a "match ACL" statement, only traffic meeting the match criteria within the ACL is forwarded to the NULL0 interface. If the NULL0 interface is specified within a stanza that does not contain a "match" statement, the match criteria is implicitly "match any."

In this example, the use of the NULL0 interface is only applicable to frames that meet the match criteria defined in the created ACL, or implicit "permit any" when no explicit match statement is listed for the stanza.

1. Configure the ACLs.

```
device(config)# ip access-list standard Jules
device(conf-ipacl-std)# permit 192.168.0.0 255.255.255.0
device(conf-ipacl-std)# deny 192.168.1.1 255.255.255.0
device(config)# ip access-list standard Vincent
device(conf-ipacl-std)# permit 192.168.2.2 255.255.255.0
```

2. Create the first stanza of the route map. (The example is using a route-map named pulp_fiction.)

```
device(config)# route-map pulp_fiction permit 10
device(config-routemap-pulp_fiction/permit/10)# match ip address acl Jules
device(config-routemap-pulp_fiction/permit/10)# set ip vrf pulp_fiction next-hop
3.3.3.3
device(config-routemap-pulp_fiction/permit/10)# set ip interface NULL0
```

3. Create the second stanza of the route map. (The example is using a route map named pulp_fiction.)

```
device(config)# route-map pulp_fiction permit 20
device(config-routemap-pulp_fiction/permit/20)# match ip address acl Vincent
device(config-routemap-pulp_fiction/permit/20)# set ip vrf pulp_fiction next-hop
3.3.3.5
```

Based on the above configuration, when address 192.168.0.0 255.255.255.0 is received, it matches stanza 10:

- If the next hop 3.3.3.3 is selected, the packet is forwarded to 3.3.3.3.
- If 3.3.3.3 is not selected by the PBR logic, the packet is sent to the next specified next-hop, which is the NULL0 interface, resulting in the traffic being dropped.
- If address 192.168.1.1 255.255.255.0 is received, since it matches the deny case of the ACL, it is denied from using the next hops specified in the route map and the traffic is forwarded according to global route table.
- If address 12.12.12.12 is received, because it meets none of the specified match criteria in either of the two stanzas, it basically falls off the end of the route map and the traffic is forwarded according to global route table.

Policy-Based Routing and NULL0 as route map default action

This example shows the use of the NULL0 interface.

In this example, the use of the NULL0 interface is only applicable to frames that meet the match criteria defined in the created ACL.

1. Configure the ACLs.

```
device(config)# ip access-list standard Jules
device(conf-ipacl-std)# permit 192.168.0.0 255.255.255.0
device(conf-ipacl-std)# deny 192.168.1.1 255.255.255.0
device(config)# ip access-list standard Vincent
device(conf-ipacl-std)# permit 192.168.2.2 255.255.255.0
```

2. Create the first stanza of the route map. (The example is using a route-map named pulp_fiction.)

```
device(config)# route-map pulp_fiction permit 10
device(config-routemap-pulp_fiction/permit/10)# match ip address acl Jules
device(config-routemap-pulp_fiction/permit/10)# set ip vrf pulp_fiction next-hop
3.3.3.3
device(config-routemap-pulp_fiction/permit/10)# set ip interface NULL0
```

3. Create the second stanza of the route map. (The example is using a route-map named pulp_fiction.)

```
device(config)# route-map pulp_fiction permit 20
device(config-routemap-pulp_fiction/permit/20)# match ip address acl Vincent
device(config-routemap-pulp_fiction/permit/20)# set ip vrf pulp_fiction next-hop
3.3.3.5
```

4. Create the third stanza, which provides the default action of the route map.

```
device(config)# route-map pulp_fiction permit 30
device(config-routemap-pulp_fiction/permit/30)# set ip interface NULL0
```

The above configuration introduces a third stanza that defines the routing desired for all frames that do not meet any of the match criteria defined by the route map.

Based on the above configuration, when address 192.168.0.0 255.255.255.0 is received, it matches stanza 10:

- If the next hop 3.3.3.3 is selected, the packet is forwarded to 3.3.3.3.
- If 3.3.3.3 is not selected by the PBR logic, the packet is sent to the next specified next-hop, which is the NULL0 interface, resulting in the traffic being dropped.
- If address 192.168.1.1 255.255.255.0 is received, since it matches the deny case of the ACL, it is denied from using the next hops specified in the route map and will be forwarded according to global route table.
- If address 12.12.12.12 is received, because it meets none of the specified match criteria in either of the first two stanzas, it reaches the third stanza. Since a no "match" statement is specified, it is an implicit "match any." The address 12.12.12.12 is forwarded to the NULL0 interface where it is dropped.

Providing the default stanza enables a mechanism whereby if any packet is received that does not meet the match criteria set by the route map, the traffic is dropped.

Policy-based routing (IPv6)

IPv6 PBR allows you to manually configure how IPv6 packets that match certain criteria can be forwarded instead of following the IPv6 Routing Table Manager (RTM) routes. This section provides configuration details of PBR policy that specifies to match and route IPv6 packets.

Configuring an IPv6 PBR with IPv6 address as the next hop

The following steps configure an IPv6 PBR by setting an IPv6 address as the next hop in the route map.

> **Note**
> This task uses access-control lists (ACLs). For details about configuring ACLs, see the "ACLs" section of this publication.

1. Enter the **configure terminal** command to enter global configuration mode.

```
device# configure terminal
```

2. Define the required IPv6 ACLs to be added to the route map.

```
device(config)# ipv6 access-list standard 99
device(conf-ip6acl-std)# permit any
```

3. Enter the **exit** command to return to global configuration mode.

```
device(conf-ip6acl-std)# exit
```

4. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all the match clauses are met.

```
device(config)# route-map test-route permit 99
```

5. Add IPv6 ACLs to match the IP address that is permitted by the ACL.

```
device(config-route-map-test-route/permit/99)# match ipv6 address acl 99
```

6. Set the IPv6 address of the next hop to which the traffic that matches a match statement in the route map must be routed.

```
device(config-route-map-test-route/permit/99)# set ipv6 next-hop
2001:db8:0:0:0:ff00:42:8329
```

7. Enter the **exit** command to return to the global configuration mode.

```
device(config-route-map-test-route/permit/99)# exit
```

8. Enable IPv6 PBR by applying the route map on an interface or on virtual interface.

- Enable IPv6 PBR by applying the route map on an interface.
  ```
  device(config)# interface ethernet 1/1
  device(conf-if-eth-1/1)# ipv6 policy route-map test-route
  ```

- Enable IPv6 PBR by applying the route map on a virtual interface.
  ```
  device(config)# interface Ve 1
  device(config-if-Ve-1)# ipv6 policy route-map test-route
  ```

The following example shows the configuration steps to configure an IPv6 PBR by setting an IPv6 address as the next hop in the route map.

```
device# configure terminal
device(config)# ipv6 access-list standard 99
device(conf-ip6acl-std)# permit any
device(conf-ip6acl-std)# exit
device(config)# route-map test-route permit 99
device(config-route-map-test-route/permit/99)# match ipv6 address acl 99
device(config-route-map-test-route/permit/99)# set ipv6 next-hop
2001:db8:0:0:0:ff00:42:8329
device(config-route-map-test-route/permit/99)# exit
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ipv6 policy route-map test-route
device(conf-if-eth-1/1)# end
```

Configuring an IPv6 PBR with NULL0 interface as the next hop

The following steps configure an IPv6 PBR by setting NULL0 interface as the next hop in the route map.

1. Enter the **configure terminal** command to enter global configuration mode.

```
device# configure terminal
```

2. Define the required IPv6 ACLs to be added to the route map.

```
device(config)# ipv6 access-list standard 99
device(conf-ip6acl-std)# permit any
```

3. Enter the **exit** command to return to global configuration mode.

```
device(conf-ip6acl-std)# exit
```

4. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all the match clauses are met.

```
device(config)# route-map test-route permit 99
```

5. Add IPv6 ACLs to match the IP address that is permitted by the ACL.

```
device(config-route-map-test-route/permit/99)# match ipv6 address acl 99
```

6. Set the next hop as NULL0 interface to send the traffic to the null interface, thus dropping the packet instead of forwarding it.

```
device(config-route-map-test-route/permit/99)# set ipv6 interface null0
```

7. Enter the **exit** command to return to the global configuration mode.

```
device(config-routemap-test-route/permit/99)# exit
```

8. Enter configuration mode on the interface where you want to enable PBR by applying the route map.

```
device(config)# interface ethernet 1/1
```

9. Enable policy-based routing on the interface and specify the route map to be used.

```
device(conf-if-eth-1/1)# ipv6 policy route-map test-route
```

The following example shows the configuration steps to configure an IPv6 PBR to send the IPv6 traffic to the null interface, thus dropping the traffic instead of forwarding it.

```
device# configure terminal
device(config)# ipv6 access-list standard 99
device(conf-ip6acl-std)# permit any
device(conf-ip6acl-std)# exit
device(config)# route-map test-route permit 99
device(config-route-map-test-route/permit/99)# match ipv6 address acl 99
device(config-route-map-test-route/permit/99)# set ipv6 interface null0
device(config-route-map-test-route/permit/99)# exit
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ipv6 policy route-map test-route
device(conf-if-eth-1/1)# end
```

Configuration examples for policy based routing

This section presents configuration examples for configuring and applying a PBR policy.

Basic example of IPv6 PBR

The following commands configure and apply an IPv6 PBR policy that routes HTTP traffic received on a virtual routing interface.

```
device# configure terminal
device(config)# ipv6 access-list standard 99
device(conf-ip6acl-std)# permit any
device(conf-ip6acl-std)# exit
device(config)# route-map test-route permit 99
device(config-route-map-test-route/permit/99)# match ipv6 address acl 99
device(config-route-map-test-route/permit/99)# set ipv6 next-hop
2001:db8:0:0:0:ff00:42:8329
device(config-route-map-test-route/permit/99)# exit
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ipv6 policy route-map test-route
device(conf-if-eth-1/1)# end
```

Port MAC Security

Port MAC security overview

Port MAC security (PMS) feature allows you to configure the device to learn a limited number of secure MAC addresses on an interface. The interface forwards only packets with source MAC addresses that match these secure addresses.

The secure MAC addresses can be specified statically or learned dynamically. If the device reaches the maximum limit for the number of secure MAC addresses allowed on the interface and if the interface receives a packet with a source MAC address that is different from any of the secure learned addresses, it is considered a security violation.

If a violation occurs, the switch responds according to one of three modes, as summarized in the following table.

Table 22: Switch responses to security violations

Response	Description
Shutdown	The default. The physical port is shut immediately and drops all traffic. A RASlog, SNMP trap, or both is sent. All logical interface operations on the physical port are disabled.
Restrict	Traffic in violation is silently dropped until the number of secure address configures drops below the maximum. Learning is disabled. All logical interface operations on the physical port are disabled.

> **Note**
> If a source MAC address is learned on one secured port, and if the same MAC address ingresses on another secured port, a MAC move is allowed and is not considered a violation.

To avoid having to intervene manually every time a port-security violation forces an interface into the shutdown state, the user can enable autorecovery for port security violations. A recovery interval is configured in seconds. After this time period the port transitions automatically to the operational state.

An age limit can be set globally for all secure addresses on a port. This feature effectively removes inactive secure addresses.

There are three types of secure MAC addresses that are used in port MAC security:

- Static MAC address: These are the secure MAC addresses that are manually configured using the **switchport port-security mac-address** command. Static MAC addresses persist even if the port goes down; or after the device reboots, provided the config is saved. When static MAC address is configured on an access secure port, the MACs qualify for access VLANs, but on a trunk port, the VLAN must be specified.
- Dynamic MAC address: These are the secure MAC addresses that the device learns automatically. Dynamically learned MAC address does not persist if the port goes down.
- Sticky MAC address: These are the secure MAC addresses that are learned dynamically but are added automatically as static MAC addresses. When sticky MAC learning is enabled on a secured port, the interface converts all the dynamic secure MAC addresses, including those that were dynamically learned before sticky learning was enabled, to sticky secure MAC addresses. All the subsequent sets of dynamically learned MAC addresses will also be converted to sticky secure MAC addresses. If sticky MAC learning is disabled on a secure port, all the sticky MAC addresses will be converted back to dynamically learned MAC addresses. Similar to the static MAC address, sticky MAC addresses persist even if the port goes down; or if the device reboots, provided the config is saved.

If a MAC address already learned on a secured port is ingressing on a nonsecured port or through another secured port, this is not considered a security violation. In this case the MAC move occurs if the MAC is learned dynamically. If the MAC is static or sticky, then the MAC move does not occur. However, the traffic will still be switched according to the destination MAC.

> **Note**
> Secure MAC addresses age out based on the device MAC age value that is configured for the device.

> **Note**
> The maximum MAC address limit for sticky MAC address and static MAC address depends on the device limit. For dynamically learned MAC addresses, the maximum limit is 8192 per port.

Port MAC security violation

A security violation occurs when the maximum limit for the number of secure MAC addresses allowed on the interface is exceeded.

When a security violation occurs, by default, the port shuts down.

> **Note**
> Refer to Table 22 on page 86.

When the port shuts down after security violation, an administrator can explicitly bring up the interface or a shutdown timer can be configured. After the configured shutdown time, the interface automatically comes up and the port security configuration remains configured on the port.

> **Note**
> When the device reboots after port shutdown due to security violation, the ports come up in the shutdown state.

Auto recovery for port MAC security violation

Auto recovery for port MAC security violation can be configured to bring up a port that is forced to shut down after a security violation by specifying a shutdown time.

The shutdown time serves as the recovery interval, providing an option to bring up a port within a configured time without any manual intervention. The shutdown time can be configured by means of the **switchport port-security shutdown-time** command. The shutdown and no-shutdown processes initiated as part of the port violation action is independent of the shutdown process explicitly initiated by an administrator on the same port on which port MAC security is enabled.

> **Note**
> Refer to Table 22 on page 86.

Port MAC security configuration guidelines and considerations

Note the following guidelines and restrictions for configuring port security:

- A port mode change is not allowed when port security is enabled on the interface.
- If a port-security-based change occurs when a port is shut down, the shutdown timer is not triggered. Consequently, the user must restore the full functionality of the port.
- When port security causes a port to be shut down and the user manually changes the shutdown time, the shutdown timer is reset and the timer starts with the new shutdown time.
- Static MAC addresses cannot be configured on a secure port. They must be configured as secure MAC addresses on the secure port.

Configuring port MAC security

The following steps are the common operations that you will need to perform for configuring port MAC security.

1. Enter the **configure terminal** command to enter global configuration mode.
   ```
   device# configure terminal
   ```
2. Enter the interface configuration mode to configure interface-specific administrative features for port MAC security.
   ```
   device(config)# interface Ethernet 3/2
   ```
3. Define the interface in Layer 2 mode to set the switching characteristics of the Layer 2 interface.
   ```
   device(conf-if-eth-3/2)# switchport
   ```
   All Layer 2 interfaces are mapped to default VLAN 1 and the interface is set to access mode. For changing the interface configuration mode to trunk or changing the default VLAN mapping, use additional **switchport** commands.

4. Enable port MAC security on the interface.

```
device(conf-if-eth-3/2)# switchport port-security
```

5. Set the maximum number of secure MAC addresses for an interface.

```
device(conf-if-eth-3/2)# switchport port-security max 10
```

For dynamically learned MAC addresses, the maximum limit is 8192 per port which is also the default value.

6. Specify the auto recovery time for port security violation.

```
device(conf-if-eth-3/2)# switchport port-security shutdown-time 4
```

7. Specify secure MAC address.

```
device(conf-if-eth-3/2)# switchport port-security mac-address 0000.00eb.2d14 vlan 2
```

8. Enable sticky MAC learning on the port to convert the dynamically learned MAC addresses to sticky secure MAC addresses.

```
device(conf-if-eth-3/2)# switchport port-security sticky
```

9. Configure port security with sticky MAC address.

```
device(conf-if-eth-3/2)# switchport port-security sticky mac-address 0000.0018.747C
vlan 5
```

The following example shows the steps to configure port MAC security.

```
device# configure terminal
device(config)# interface Ethernet 3/2
device(conf-if-eth-3/2)# switchport
device(conf-if-eth-3/2)# switchport port-security
device(conf-if-eth-3/2)# switchport port-security max 8192
device(conf-if-eth-3/2)# switchport port-security shutdown-time 4
device(conf-if-eth-3/2)# switchport port-security mac-address 0000.00eb.2d14 vlan 2
device(conf-if-eth-3/2)# switchport port-security sticky mac-address 0000.0018.747C vlan 5
device(conf-if-eth-3/2)# switchport port-security violation shutdown
```

Displaying port MAC security information

When port MAC security is enabled, various **show** commands can be used to display information about port security and secure MAC addresses.

You can display the following information about the port MAC security feature:

- The details of port MAC security configured on the device
- The port MAC security configuration details
- The port MAC security settings for an individual port
- The secure MAC addresses configured on the device

Displaying port MAC security details on the device

To display the port MAC security configured on a particular interface, enter the following command:

```
device# configure terminal
device(config)# interface Ethernet 3/2
device(conf-if-eth-3/2)# do show run interface Ethernet 3/2
interface Ethernet 3/2
switchport
switchport mode trunk
switchport port-security
```

```
switchport port-security max 10
switchport port-security mac-address 3200.1110.0002 vlan 250
switchport trunk allowed vlan add 250
switchport trunk tag native-vlan
no shutdown
```

Displaying port MAC security configuration details

To display the port MAC security configuration details across ports on the device, enter the following command:

```
device(conf-if-eth-3/2)# do show port-security
Secure      MaxSecureAddr  CurrentAddr  StaticSec  Violated  Action    Sticky
Port          (count)        (count)      (count)
Eth 3/2        10             0            1          No      Shutdown  No
```

Displaying port MAC security settings for an individual port

To display the statistics of the port MAC security configured for an interface, enter the following command:

```
device(conf-if-eth-3/2)# do show port-security interface ethernet 3/2
Port Security            : Enabled
Port Status              : Up
Violation Mode           : Shutdown
Violated                 : No
Sticky Enabled           : No
Maximum MAC addresses    : 10
Total MAC addresses      : 0
Configured MAC addresses : 1
Last violation time      :
Shutdown time (in Minutes) : 0
```

Displaying secure MAC addresses information

To list the secure MAC addresses configured on the device, enter the following command.

```
device(conf-if-eth-3/2)# do show port-security addresses
                Secure Mac Address Table
-----------------------------------------------------------------
Vlan      Mac-address          Type                    Ports
250       3200.1110.0002       Secure-Static           Eth 3/2
```

802.1x authentication

802.1X authentication overview

The IEEE 802.1X standard is designed to govern the authentication of devices attached to LAN ports. The 802.1X protocol defines a port-based authentication algorithm involving network data communication between client-based supplicant software, an authentication database on a server, and the authenticator device. Using 802.1X authentication, you can configure a device to grant access to a port based on information supplied by a client to an authentication server.

When a user logs on to a network that uses 802.1X authentication, the device grants (or does not grant) access to network services after the user is authenticated by an authentication server. The user-based authentication in 802.1X authentication provides an alternative to granting network access based on a user's IP address, MAC address, or subnetwork.

The Extreme implementation of 802.1X authentication supports the following RFCs:

- RFC 2284: PPP Extensible Authentication Protocol (EAP)
- RFC 2865: Remote Authentication Dial In User Service (RADIUS)

- RFC 2869: RADIUS Extensions

> **Note**
> SNMP is not supported for 802.1X authentication.

Device roles in an 802.1X configuration

The 802.1X standard defines the roles of client/supplicant, authenticator, and authentication server in a network.

The client (known as a supplicant in the 802.1X standard) provides username and password information to the authenticator. The authenticator sends this information to the authentication server. Based on the client's information, the authentication server determines whether the client can use services provided by the authenticator. The authentication server passes this information to the authenticator, which then provides services to the client, based on the authentication result.

The following figure illustrates these roles.
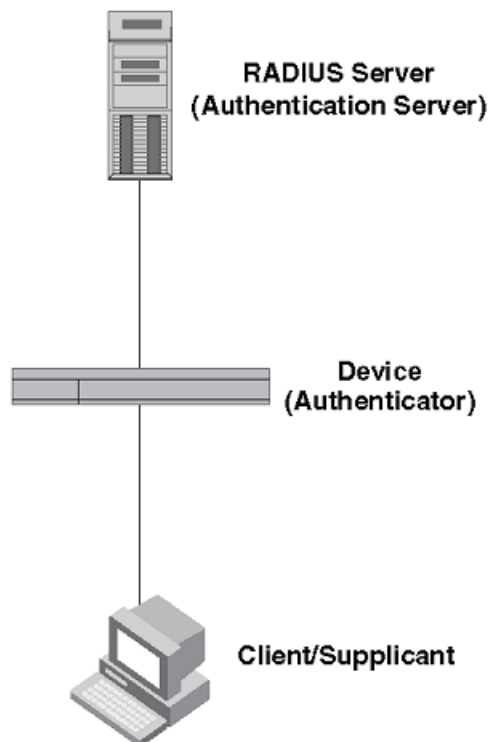
Figure 5: Authenticator, client/supplicant, and authentication server in an 802.1X configuration

Authenticator: The device that controls access to the network. In an 802.1X configuration, the device serves as the authenticator. The authenticator passes messages between the client and the

authentication server. Based on the identity information supplied by the client, and the authentication information supplied by the authentication server, the authenticator either grants or does not grant network access to the client.

Client/supplicant: The device that seeks to gain access to the network. Clients must be running software that supports the 802.1X standard (for example, the Windows 7 operating system). Clients can either be directly connected to a port on the authenticator, or can be connected by way of a hub.

Authentication server: The device that validates the client and specifies whether or not the client may access services on the device. Extreme supports authentication servers running RADIUS.

Communication between the devices

For communication between the devices, 802.1X uses the Extensible Authentication Protocol (EAP), defined in RFC 2284. The 802.1X standard specifies a method for encapsulating EAP messages so that they can be carried over a LAN. This encapsulated form of EAP is known as EAP over LAN (EAPOL). The standard also specifies a means of transferring the EAPOL information between the client/ supplicant, authenticator, and authentication server.

EAPOL messages are passed between the Port Access Entity (PAE) on the supplicant and the authenticator. The following figure shows the relationship between the authenticator PAE and the supplicant PAE.
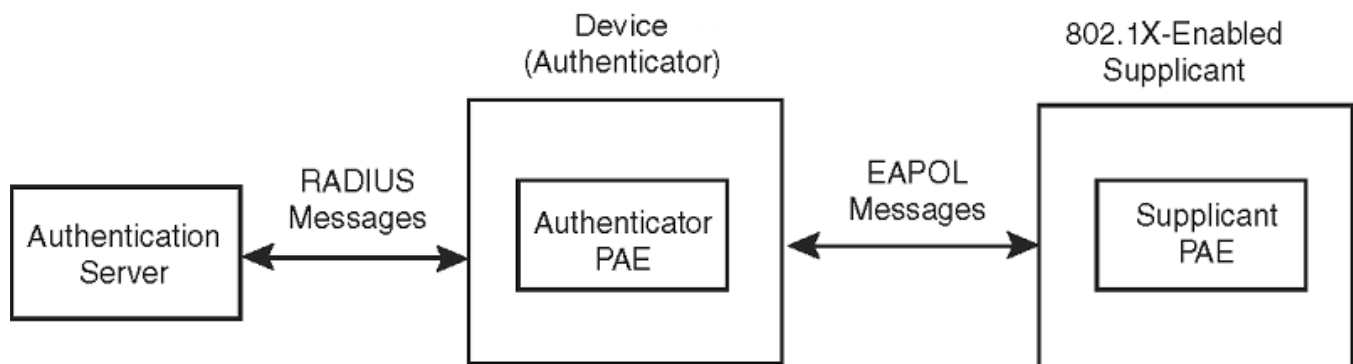
Figure 6: Authenticator PAE and supplicant PAE

Authenticator PAE: The authenticator PAE communicates with the supplicant PAE, receiving identifying information from the supplicant. Acting as a RADIUS client, the authenticator PAE passes the supplicant information to the authentication server, which decides whether the supplicant can gain access to the port. If the supplicant passes authentication, the authenticator PAE grants it access to the port.

Supplicant PAE: The supplicant PAE supplies information about the client to the authenticator PAE and responds to requests from the authenticator PAE. The supplicant PAE can also initiate the authentication procedure with the authenticator PAE, as well as send log off messages.

Controlled and uncontrolled ports

A physical port on the device used with 802.1X authentication has two virtual access points: a controlled port and an uncontrolled port. The controlled port provides full access to the network. The uncontrolled port provides access only for EAPOL traffic between the client and the authenticator. When a client is successfully authenticated, the controlled port is opened to the client. The following figure illustrates this concept.
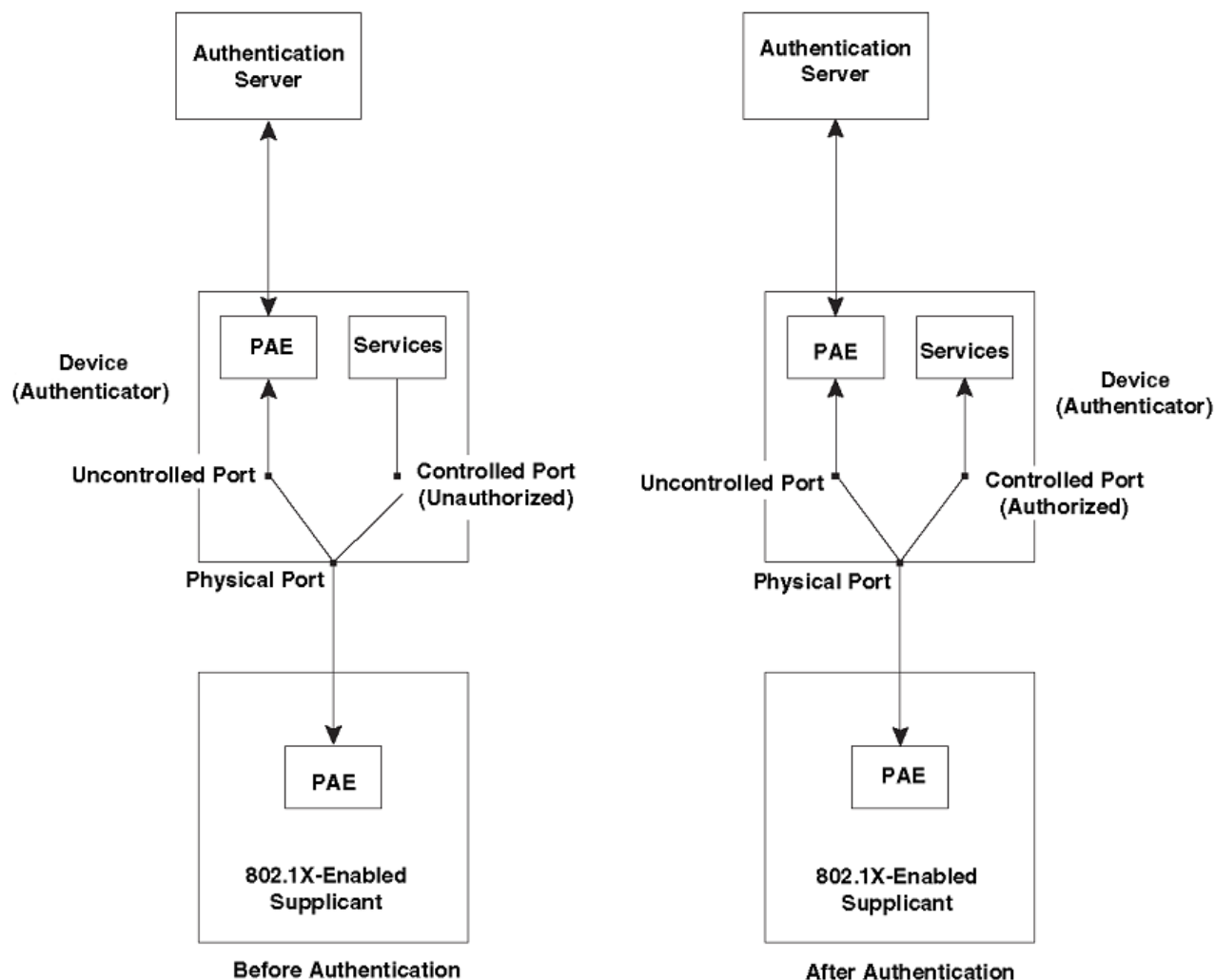
Figure 7: Controlled and uncontrolled ports before and after client authentication

Before a client is authenticated, only the uncontrolled port on the authenticator is open. The uncontrolled port allows only EAPOL frames to be exchanged between the client and the authenticator. The controlled port is in the unauthorized state and allows no traffic to pass through.

During authentication, EAPOL messages are exchanged between the supplicant PAE and the authenticator PAE, and RADIUS messages are exchanged between the authenticator PAE and the authentication server. If the client is successfully authenticated, the controlled port becomes authorized for that client, and traffic from the client can flow through the port normally. When a client connected to the port is successfully authenticated, client is authorized to send traffic through controlled port until the client logs off.

Message exchange during authentication

The following figure illustrates a sample exchange of messages between an 802.1x-enabled client, a device acting as authenticator, and a RADIUS server acting as an authentication server.
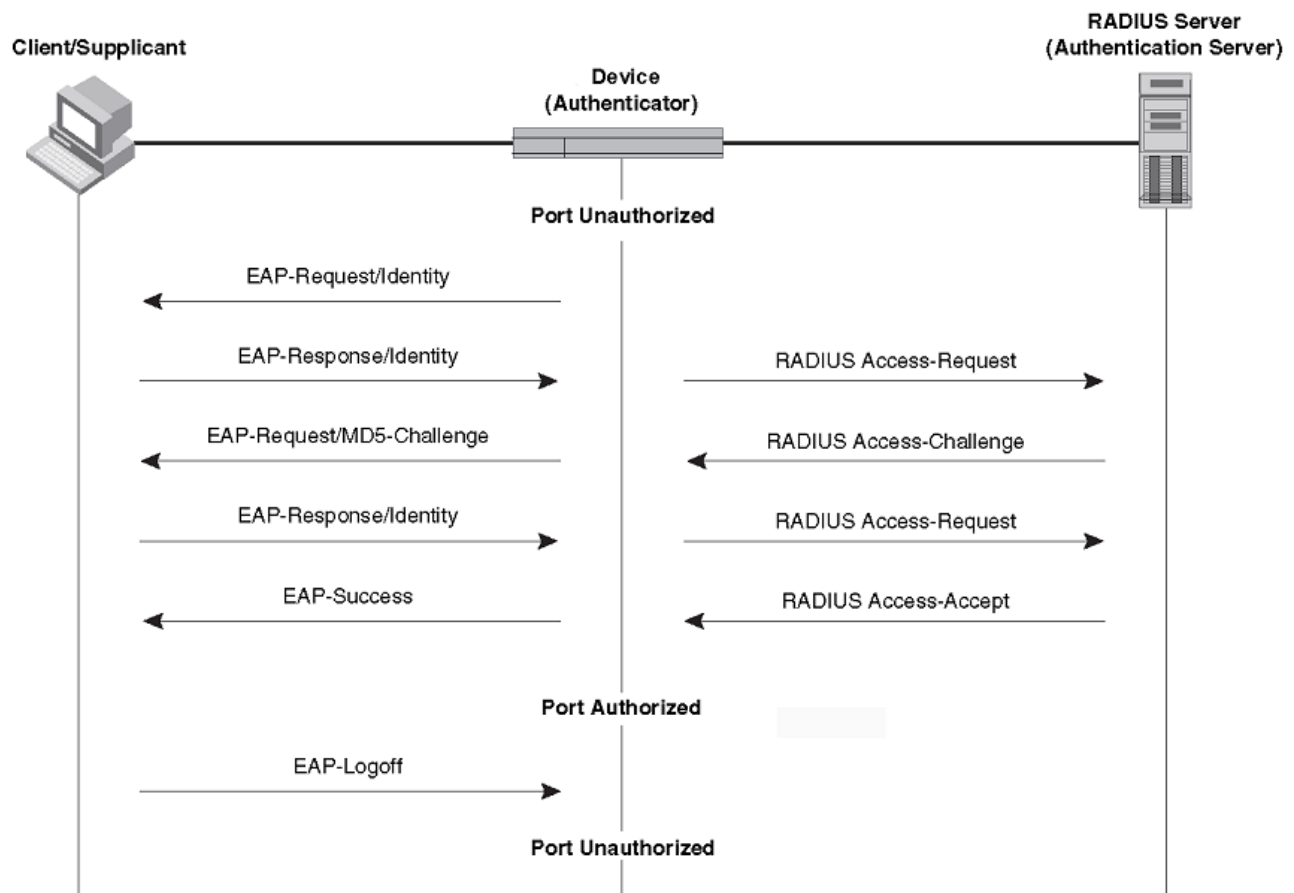
Figure 8: Message exchange between client/supplicant, authenticator, and authentication server

In this example, the authenticator (the device) initiates communication with an 802.1x-enabled client. When the client responds, it is prompted for a username and password. The authenticator passes this information to the authentication server, which determines whether the client can access services provided by the authenticator. When the client is successfully authenticated by the RADIUS server, the client is authorized to use services provided by the authenticator. When the client logs off, the port becomes unauthorized for that client.

If a client does not support 802.1x, authentication cannot take place. The device sends EAP-Request/ Identity frames to the client, but the client does not respond to them.

When a Client that supports 802.1X attempts to gain access through a non-802.1X-enabled port, it sends an EAP start frame to the device. When the device does not respond, the client considers the port to be authorized, and starts sending normal traffic.

The Extreme 802.1x implementation supports dynamic VLAN assignment. If one of the attributes in the Access-Accept message sent by the RADIUS server specifies a VLAN identifier, and this VLAN is available on the device, the client port is moved from its default VLAN to the specified VLAN. When the

client disconnects from the network, the port is placed back in its default VLAN. For more information, refer to Dynamic VLAN assignment for 802.1x ports.

The Extreme 802.1x implementation supports dynamically applying an IP ACL or MAC ACL to a port, based on information received from the authentication server. When the client disconnects from the network, the assigned ACLs will be removed from the port.

Authentication of multiple clients connected to the same port

Devices support 802.1X authentication for ports with more than one client connected to them. The following figure illustrates a sample configuration where multiple clients are connected to a single 802.1X port.
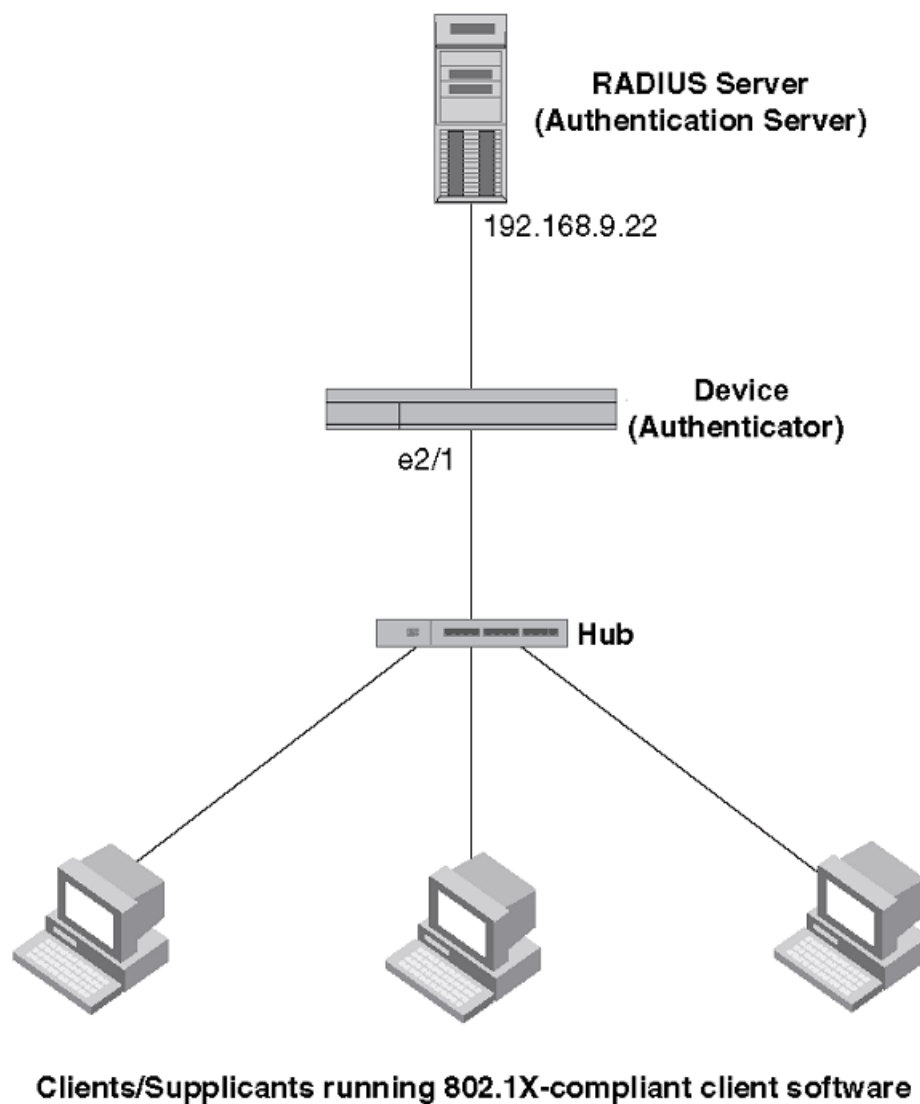
Figure 9: Multiple clients connected to a single 802.1X-enabled port

If there are multiple clients connected to a single 802.1X-enabled port, the device authenticates each of them individually. Each client's authentication status is independent of the others, so that if one

authenticated client disconnects from the network, it has no effect on the authentication status of any of the other authenticated clients. The number of clients supported at the system level is 255.

By default, traffic from clients that cannot be authenticated by the RADIUS server is dropped.

How 802.1x multiple client authentication works

When multiple clients are connected to a single 802.1x-enabled port on a router (as in Authentication of multiple clients connected to the same port on page 96), 802.1x authentication is performed in the following ways.

1. One of the 802.1x-enabled clients attempts to log into a network in which a device serves as an Authenticator.
2. The device performs 802.1x authentication for the client. Messages are exchanged between the device and the client, and between the device and the Authentication Server (RADIUS server). The result of this process is that the client is either successfully authenticated or not authenticated, based on the username and password supplied by the client.
3. If the client is successfully authenticated, traffic from the client is forwarded normally.
4. When the client disconnects from the network, the device marks the client as unauthorized and the status is displayed in the output of `show dot1x session-info` command with the `interface ethernet` options. This does not affect the authentication status (if any) of the other clients connected on the port.

RADIUS attributes for authentication

RADIUS attributes are used to define specific authentication, authorization, and accounting (AAA) elements in a user profile, which is stored in the RADIUS server. When a client successfully completes the EAP authentication process, the authentication server (the RADIUS server) sends the authenticator device a RADIUS Access-Accept message that grants the client access to the network. The RADIUS Access-Accept message contains attributes set for the user in the user's access profile on the RADIUS server.

Many functions, such as dynamic VLAN assignment and dynamic IP ACL and MAC filter assignment, are based on the RADIUS attributes. Extreme devices support the following RADIUS attributes for 802.1X authentication:

* Username (1): RFC 2865
* Filter-Id (11): RFC 2865
* Tunnel-Type (64): RFC 2868
* Tunnel-Medium-Type (65): RFC 2868
* Tunnel-Private-Group-Id (81): RFC 2868

Support for the RADIUS user-name attribute in Access-Accept messages

Authentication-enabled ports support the RADIUS user-name (type 1) attribute in the Access-Accept message returned during authentication.

When sFlow is enabled on the port, sFlow samples taken from the interface include the username of 802.1X client based on the source MAC address of the sFlow sample. For example, when the user-name

attribute is sent in the Access-Accept message, it is then available for display in sFlow sample messages sent to a collector, and in the output of some **show dot1x** commands, such as **show dot1x session-info**.

To enable the user-name attribute, add the following attribute on the RADIUS server.

Table 23: RADIUS user-name attribute details

Attribute name	Type	Value
user-name	1	*name* (string)

Dynamic VLAN assignment for 802.1X ports

The Extreme 802.1X implementation supports assigning a port to a VLAN dynamically, based on information received from an authentication server (RADIUS server).

When a client or supplicant successfully completes the EAP authentication process, the authentication server (RADIUS server) sends the authenticator (the device) a RADIUS Access-Accept message that grants the client access to the network. The RADIUS Access-Accept message contains attributes set for the user in the user's access profile on the RADIUS server.

If one of the attributes in the Access-Accept message specifies a VLAN identifier (ID), and this VLAN is available on the device, the client's port is moved from its default VLAN to the specified VLAN. When the client disconnects from the network, the port is placed back in its default VLAN.

To enable 802.1X VLAN ID support on the device, you must add the following attributes to a user's profile on the RADIUS server.

Table 24: RADIUS attributes for dynamic VLAN assignment

Attribute name	Type	Value
Tunnel-Type	064	13 (decimal) - VLAN
Tunnel-Medium-Type	065	6 (decimal) - 802
Tunnel-Private-Group-ID	081	*vlan-number* (decimal).

The device reads the attributes as follows:

- All three VLAN ID attributes (Tunnel-Private-Group-ID, Tunnel-Type, and Tunnel-Medium-Type) must be present in the response from the RADIUS server for VLAN processing.

- If the Tunnel-Type or Tunnel-Medium-Type attributes (or both) are not present, then the client is moved to the unauthorized state displaying an error message on the device.

- If the Tunnel-Type or Tunnel-Medium-Type attributes in the Access-Accept message have the values specified in the table, but there is no value specified for the Tunnel-Private-Group-ID attribute, the client will not become authorized.

- When the device receives the value specified for the Tunnel-Private-Group-ID attribute, it checks whether the *vlan-ID* matches the VLAN configured on the device. If there is a VLAN on the device that matches the *vlan-ID*, then the client's port is placed in the VLAN with an that ID corresponds to the VLAN ID.

Considerations for dynamic VLAN assignment in an 802.1X multiple client configuration

The port must be the switch port for allowing dynamic VLAN assignment and the corresponding VLAN must be preconfigured on the device. If the RADIUS Access-Accept message specifies the ID of a VLAN that does not exist on the device, then it is considered an authentication failure. If the port is not already a member of a RADIUS-specified VLAN, and the RADIUS Access-Accept message specifies the ID of a valid VLAN on the device, then the port is placed in that VLAN. When the client disconnects from the network, the port is moved out of the VLAN.

The client port is moved to the specified VLAN as tagged or untagged depending on the VLAN port mode (access, trunk, or hybrid). When multiple clients connect to the port with different VLANs, the VLAN is applied based on the port mode, which is either access or trunk.

In the case of access mode, the VLAN ID that is received for the first client is applied on the port. The subsequent clients authenticated with different VLANs are rejected. The port's VLAN membership is not changed. However, for trunk ports, multiple VLANs can be tagged.

> **Note**
> Dynamically assigned VLANs are not displayed in the running-config. So, you must ensure that VLAN is not manually configured on the corresponding 802.1X authentication-enabled port.

Dynamic ACLs and MAC address filters in authentication

After successful authentication, different network policies can be applied to restrict the way the network resources are accessed by the client. The 802.1X authentication implementation supports dynamically applying an IP ACL to a port, based on information received from the authentication server. The 802.1X authentication also supports dynamic assignment of MAC ACLs to a port.

> **Note**
> ACL must not be manually applied to an 802.1X authentication-enabled port.

When a client or supplicant is authenticated, the authentication server (the RADIUS server) sends the authenticator (the device) a RADIUS Access-Accept message containing the Filter-Id (type 11) attribute, the device can use information in the attribute to apply an IP ACL or MAC ACL to the authenticated port. This IP ACL or MAC ACL applies to the port for as long as the client is connected to the network. The IP ACL or MAC ACL is removed from the corresponding port when the client logs out, or the port goes down.

The ACL IDs received in the Radius Access-Accept message for the first authenticated client is applied to the port. The subsequent authenticated clients that receive the same ACL IDs will be authorized. If the subsequent clients receive different ACL IDs, they will be considered unauthorized. If all the clients are logged out due to a log-off message from the clients, the assigned ACL ID set is removed from the port when the last client logs out.

The device uses information in the Filter-Id attributes that can specify existing IP ACL or MAC ACL configured on the device. IP ACL or MAC ACL with the specified ACL name is applied to the port.

> **Note**
> Only IPv4 ACL is supported and IPv6 ACL binding is not supported.

ACL bind fails in the following scenarios:

- The ACL is not configured in the device.
- The port is already applied with the same ACL type in the same direction but different ACL ID.
- The port is not a switch port and the MAC type ACL is sent from the RADIUS server for that port.
- The ACL type sent from the RADIUS server does not match with the ACL type configured in the device.
- On the same port, multiple clients are connected with different ACLs. In this case, the ACL that is sent for the first client is applied on the port. The other clients with different ACLs are rejected.

> **Note**
> Dynamically assigned ACLs are not displayed in the running-config.

Dynamically applying existing ACLs or MAC ACL

When a port is authenticated using 802.1X security, an IP ACL or MAC ACL that exists in the running configuration on the device can be dynamically applied to the port. To do this, you configure the Filter-ID (type 11) attribute on the RADIUS server. The Filter-Id attribute specifies the name of the IP ACL or MAC ACL.

The following table shows the syntax for configuring the Filter-Id attribute to refer to an IP ACL or MAC ACL.

Table 25: Syntax for Filter-Id attribute

Value	Description
ip.*name*.in	Applies the specified named ACL to the 802.1X authenticated port in the inbound direction.
ip.*name*.out	Applies the specified named ACL to the 802.1X authenticated port in the outbound direction.
mac.*name*.in	Applies the specified MAC ACL to the 802.1X authenticated port in the inbound direction.

> **Note**
> - The `name` variable in the Filter-Id attribute is case-sensitive.
> - Dynamic IP ACL filters are supported for the inbound and outbound directions.
> - MAC ACLs are supported only for the inbound direction. Outbound MAC ACLs are not supported.
> - The RADIUS server sends the ACL name in the Filter-Id attribute in the following form: <ip/mac>.<acl_name>.<in/out>
> - Only one ACL ID per ACL type is allowed in each direction.

Strict security mode for dynamic filter assignment

By default, dynamic filter assignment operates in strict security mode. When strict security mode is enabled, authentication for a port fails if the Filter-Id attribute contains invalid information to implement

the IP ACLs or MAC ACLs. You can manually disable the strict security mode using the **no filter-strict-security** command in the interface configuration mode.

When strict security mode is enabled:

- If the Filter-Id attribute in the Access-Accept message contains a value that does not refer to an existing filter (that is, a MAC ACL or IP ACL configured on the device), then the client will not be authorized, regardless of any other information in the message (for example, if the Tunnel-Private-Group-ID attribute specifies a VLAN on which to assign the port).

When strict security mode is disabled:

- If the Filter-Id attribute in the Access-Accept message contains a value that does not refer to an existing filter (that is, a MAC ACL or IP ACL configured on the device), then the client remains authorized and no filter is dynamically applied to it.

802.1x readiness check

The 802.1X readiness check audits all the ports for 802.1X activity and displays information about the devices with 802.1X-supported ports. The 802.1X readiness check can be used to establish whether the devices connected to the ports are 802.1X-capable.

The 802.1X readiness check is allowed on all ports that can be configured for 802.1X. The 802.1X readiness check is not available on a port that is configured by the **dot1x port-control force-unauthorized** command.

When you execute the **dot1x test eapol-capable** command on an 802.1X-enabled port, and the link comes up, the port queries the connected client about its 802.1X capability. When the client responds with a notification packet, it is 802.1X-capable. A RASLog message is generated if the client responds within the timeout period. If the client does not respond to the query, the client is not 802.1X-capable, and a syslog message is generated indicating the client is not EAPOL-capable.

Follow these guidelines to enable the 802.1X readiness check on the device:

- The 802.1X readiness check is typically used before 802.1X is enabled on the device.
- 802.1X authentication cannot be initiated while the 802.1X readiness check is in progress.
- The 802.1X readiness check cannot be initiated while 802.1X authentication is active.
- 802.1X readiness can be checked on a per-interface basis.
- The 802.1X readiness check for all interfaces at once is not supported.
- The 802.1X test timeout is shown in the output of the **show dot1x** command.

802.1X authentication enablement

By default, 802.1X authentication is disabled on the device. To enable 802.1X authentication, you must initialize 802.1X authentication globally and then enable 802.1X authentication on a specific interface.

The **dot1x enable** command in the global configuration mode initializes 802.1X authentication globally on all ports. After which, you can enable 802.1x authentication on a specific interface using the **dot1x authentication** command in interface configuration mode.

Port control for authentication

To activate authentication on an 802.1X-enabled interface, you must specify the kind of port control to be used on the interface.

The port control type can be one of the following:

- force-authorized: The controlled port is placed unconditionally in the authorized state, allowing all traffic. This is the default state for ports on the device.
- force-unauthorized: The controlled port is placed unconditionally in the unauthorized state.
- auto: The controlled port is unauthorized until authentication takes place between the client and the authentication server. Once the client passes authentication, the client is authorized to send traffic through that port. Auto is the default port control type used when 802.1X authentication is enabled on the port.

> **Note**
> Before activating the authentication on a port, you must remove the configured static ACL and static VLANs, if any, from the port.

> **Note**
> Do not configure ACLs or VLANs through the CLI manually on the authentication-enabled port.

802.1x client reauthentication options

There are a number of 802.1x client reauthentication options.

Periodic reauthentication

You can configure the device to periodically reauthenticate clients connected to 802.1x-enabled interfaces. When periodic reauthentication is enabled using the **dot1x reauthentication** command, the device reauthenticates the clients every 3,600 seconds by default. The **dot1x timeout re-authperiod** command resets the reauthentication interval, which takes precedence over the default interval.

Manual reauthentication of a port

When periodic reauthentication is enabled, the device reauthenticates clients connected to an 802.1X-enabled interface every 3,600 seconds (or the time specified by the **dot1x timeout re-authperiod** command) by default. You can also manually reauthenticate clients connected to a specific port using the **dot1x reauthenticate** command in the privileged EXEC mode.

Quiet period for reauthentication

If the device is unable to authenticate the client, the device waits for a specified amount of time before trying again. The amount of time the device remains idle between a failed authentication and a reauthentication attempt is specified with the **dot1x timeout quiet-period** command.

Retransmission information for EAP-Request/Identity frames

There are a number of configurable retransmission options for Extensible Authentication Protocol (EAP) request or identity frames.

Retransmission interval for EAP-Request/Identity frames

When the device sends a client an EAP-Request/Identity frame, it expects to receive an EAP-Response/Identity frame from the client. If the client does not send back an EAP-Response/Identity frame, the device waits a specified amount of time and then retransmits the EAP-Request/Identity frame. You can specify the amount of time the device waits before retransmitting the EAP-Request/Identity frame to the client. This amount of time is specified using the **dot1x timeout tx-period** command.

Retransmission timeout of EAP-Request frames to the client

Acting as an intermediary between the RADIUS authentication server and the client, the device receives RADIUS messages from the RADIUS server, encapsulates them as EAPOL frames, and sends them to the client. When the device relays an EAP-Request frame from the RADIUS server to the client, it expects to receive a response from the client within 30 seconds. If the client does not respond within the allotted time, the device retransmits the EAP-Request frame to the client. The timeout value for retransmission of EAP-Request frames to the client can be configured using the **dot1x timeout supp-timeout** command.

Retransmission limit for EAP-Request/Identity frame

If the device does not receive an EAP-Response/Identity frame from a client, the device waits 30 seconds (or the amount of time specified with the **dot1x timeout tx-period** command), and then retransmits the EAP-Request/Identity frame. By default, the device retransmits the EAP-Request/Identity frame a maximum of two times. If no EAP-Response/Identity frame is received from the client after two EAP-Request/Identity frame retransmissions, the device restarts the authentication process with the client.

You can specify from 1 through 10 frame retransmissions using the **dot1x max-req** command.

Configuring 802.1x authentication

To enable and activate 802.1X authentication, perform the following steps.

802.1x authentication requires some prerequisite tasks be performed before executing 802.1x authentication configurations at the global and interface levels. Before configuring 802.1x authentication, communication between the devices and the authentication server must be established. The following configurations must be completed before configuring 802.1X authentication:

- Configure the RADIUS server to authenticate access to the device. The **radius-server host** command adds the RADIUS server to the device as the authentication server. This command can be repeated for additional servers. The **radius-server host** command attempts to connect to the first RADIUS server. If the RADIUS server is not reachable, the next RADIUS server is contacted. If the

RADIUS server is contacted and the authentication fails, the authentication process does not check for the next server in the sequence.

> **Note**
>
> If multiple RADIUS servers are configured, the recommended configuration for RADIUS server retries is 2.

1. (Optional) Enable the 802.1X readiness check on the device to determine if the devices connected to the switch ports are 802.1X-capable.

   ```
   device# dot1x test eapol-capable interface ethernet 1/1
   device# 2016/07/18-00:49:03, [DOT1-1012], 5006, M2 | Active | DCE, INFO, sw0,
   DOT1X_PORT_EAPOL_CAPABLE: Peer connected to port Ethernet 1/1 is EAPOL capable.
   ```

2. Enter global configuration mode.

   ```
   device# configure terminal
   ```

3. Enable 802.1X authentication globally.

   ```
   device(config)# dot1x enable
   ```

   If you globally disable 802.1X authentication, then all interface ports with 802.1X authentication enabled, automatically switch to force-authorized port control mode.

4. Enter interface configuration mode to configure interface-specific administrative features for 802.1X authentication.

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

5. Enable 802.1X authentication on a specific interface port.

   ```
   device(conf-if-eth-1/1)# dot1x authentication
   ```

6. Enter the **dot1x port-control auto** command to set the controlled port in the unauthorized state until authentication takes place between the client and the authentication server.

   ```
   device(conf-if-eth-1/1)# dot1x port-control auto
   ```

   The action activates authentication on an 802.1X-enabled interface. Once the client passes authentication, the port becomes authorized for that client. The controlled port remains in the authorized state for that client until the client logs off .

7. (Optional) Configure the device to periodically reauthenticate the clients connected to 802.1X-enabled interfaces at regular intervals.

   ```
   device(conf-if-eth-1/1)# dot1x reauthentication
   ```

   When you enable periodic reauthentication, the device reauthenticates the clients every 3,600 seconds by default.

8. (Optional) Configure the timeout parameters that determine the time interval for client reauthentication and EAP retransmissions using the following commands:

- Enter the **dot1x timeout re-authperiod** command to change and specify a different reauthentication interval.

  ```
  device(conf-if-eth-1/1)# dot1x timeout re-authperiod 300
  ```

- Enter the **dot1x timeout tx-period** command to change the amount of time the device should wait before retransmitting EAP-Request/Identity frames to the client.

  ```
  device(conf-if-eth-1/1)# dot1x timeout tx-period 30
  ```

- Enter the **dot1x timeout supp-timeout** command to change the amount of time the device should wait before retransmitting RADIUS EAP-Request/Challenge frames to the client.

  ```
  device(conf-if-eth-1/1)# dot1x timeout supp-timeout 30
  ```

Based on the timeout parameters, client reauthentication and retransmission of EAP-Request/ Identity frames and EAP-Request/Challenge frames is performed.

9. (Optional) Configure the maximum number of reauthentication attempts before the port goes to the unauthorized state.

```
device(conf-if-eth-1/1)# dot1x reauthMax 3
```

10. (Optional) Configure the time interval the device remains idle between a failed authentication and a reauthentication attempt.

```
device(conf-if-eth-1/1)# dot1x quiet-period 30
```

11. (Optional) Enter the **no dot1x filter-strict-security** command to authenticate the client even if the Filter-Id attribute returned by RADIUS contains invalid information.

```
device(conf-if-eth-1/1)# no dot1x filter-strict-security
```

By default, strict security mode is enabled.

Displaying 802.1x information

Various show commands can be used to display the following 802.1x-related information:

- Information about the 802.1x configuration on the device and on individual ports
- Statistics about the EAPOL frames passing through the device
- Information about 802.1x-enabled ports dynamically assigned to a VLAN
- Information about the dynamically applied MAC and IP ACLs currently active on the device
- Information about the 802.1x multiple client configuration

Enter the **show dot1x** command to display the overall state of 802.1X authentication on the system.

```
device# show dot1x

802.1X Port-Based Authentication: Enabled
PAE Capability:                    Authenticator Only
Protocol Version:              2
Auth Server:                       RADIUS
Readiness test timeout:        10
RADIUS Configuration
-------------------
Position:              1
Server Address:        10.24.65.6
Port:                  1812
```

```
Secret:                   xxxxxxxxx
Retry Interval:           5 seconds
```

Enter the **show dot1x all** command to display detailed 802.1X authentication information for all of the ports.

```
device# show dot1x all

802.1X Port-Based Authentication: Enabled
PAE Capability:                   Authenticator Only
Protocol Version:                 2
Auth Server:                      RADIUS
Readiness test timeout:           10

RADIUS Configuration
--------------------
Position:              1
Server Address:        10.20.106.144
Port:                  1812
Secret:                testing123
Retry Interval:        4 seconds
Position:              2
Server Address:        10.20.106.189
Port:                  1812
Secret:                testing123
Retry Interval:        4 seconds

802.1X info for interface Eth 1/31
----------------------------------
Port Control:          Auto
Protocol Version:      2
ReAuthentication:      Enabled
Auth Fail Max Attempts: 0
ReAuth Max:            2
Tx Period:             30 seconds
Quiet Period:          60 seconds
Supplicant Timeout:    30 seconds
Re-Auth Interval:      3600 seconds
Dynamic VLAN assigned: 50
Filter-strict-security: Enabled
IP ACL assigned (IN|OUT): IPEXT-50 | IPEXT-OUT-50
MAC ACL assigned:      mac-ext
```

Enter the **show dot1x diagnostics interface** command to display all diagnostics information for the authenticator associated with a port.

```
device# show dot1x diagnostics interface ethernet 1/2

802.1X Diagnostics for interface Eth 1/2
---------------------------------------
authEnterConnecting:               1
authEaplogoffWhileConnecting:      0
authEnterAuthenticating:           1
authSuccessWhileAuthenticating:    1
authTimeoutWhileAuthenticating:    0
authFailWhileAuthenticating:       0
authEapstartWhileAuthenticating:   0
authEaplogoffWhileAuthenticating:  0
authReauthsWhileAuthenticated:     0
authEapstartWhileAuthenticated:    0
authEaplogoffWhileAuthenticated:   0
BackendResponses:                  11
BackendAccessChallenges:           10
```

```
BackendOtherrequestToSupplicant:    11
BackendAuthSuccess:                 1
BackendAuthFails:                   0
```

Enter the **show dot1x interface** command to display state of a specified interface.

```
show dot1x interface ethernet 1/31

802.1X info for interface Eth 1/31
--------------------------------
Port Control:           Auto
Protocol Version:       2
ReAuthentication:       Enabled
Auth Fail Max Attempts: 0
ReAuth Max:             2
Tx Period:              30 seconds
Quiet Period:           60 seconds
Supplicant Timeout:     30 seconds
Server Timeout:         30 seconds
Re-Auth Interval:       3600 seconds
Dynamic VLAN assigned:  50
Filter-strict-security: Enabled
IP ACL assigned (IN|OUT): IPEXT-50 | IPEXT-OUT-50
MAC ACL assigned:       mac-ext
```

Enter the **show dot1x session-info interface** command to display information for all clients on the port .

```
device# show dot1x session-info interface ethernet 1/2

802.1X Session info for interface Eth 1/2
----------------------------------------
Mac Address: 0021.5ec6.15ce
----------------------------------------
User Name:              md5user2
Session Time:           2 secs
Terminate Cause:        Not terminated yet
Session Status:         Authorized
PAE State:              Authenticated
BE State:               Idle
VLAN:                   N/A
IP ACL (IN | OUT):      N/A | N/A
MAC ACL:                N/A
Current Id:             18
Id From Server:         17
```

Enter the **show dot1x statistics interface** command to display the statistics of a specified interface.

```
device# show dot1x statistics interface ethernet 1/2

802.1X statistics for interface Eth 1/2
-------------------------------------
EAPOL Frames Rx:                    12
EAPOL Frames Tx:                    43
EAPOL Start Frames Rx:              1
EAPOL Logoff Frames Rx:            0
EAP Rsp/Id Frames Rx:              1
EAP Response Frames Rx:            10
EAP Req/Id Frames Tx:              23
EAP Request Frames Tx:             10
Invalid EAPOL Frames Rx:           0
EAPOL Length Error Frames Rx:      0
```

```
EAPOL Last Frame Version Rx:        1
Invalid EAP Frames Rx:              0
EAP Length Error Frames Rx:         0
EAPOL Last Frame Src:               0021.5ec6.15ce
```

Configuring Remote Server Authentication

Remote server authentication overview on page 109
Configuring remote server authentication on page 110

Remote server authentication overview

The software supports various protocols to provide external Authentication, Authorization, and Accounting (AAA) services for devices. Supported protocols include the following:

- RADIUS — Remote authentication dial-in user service
- LDAP/AD — Lightweight Directory Access Protocol using Microsoft Active Directory (AD) in Windows
- TACACS+ — Terminal access controller access-control system plus

When configured to use a remote AAA service, the device acts as a network access server client. The device sends all authentication, authorization, and accounting (AAA) service requests to the remote RADIUS, LDAP, or TACACS+ server. The remote AAA server receives the request, validates the request, and sends a response back to the device.

The supported management access channels that integrate with RADIUS, TACACS+, or LDAP include serial port, Telnet, or SSH.

When configured to use a remote RADIUS, TACACS+, or LDAP server for authentication, a device becomes a RADIUS, TACACS+, or LDAP client. In either of these configurations, authentication records are stored in the remote host server database. Login and logout account name, assigned permissions, and time-accounting records are also stored on the AAA server for each user.

Extreme recommends that you configure at least two remote AAA servers to provide redundancy in the event of failure. For each of the supported AAA protocols, you can configure up to five external servers on the device. Each device maintains its own server configuration.

Login authentication mode

The authentication mode is defined as the order in which AAA services are used on the device for user authentication during the login process. The software supports two sources of authentication: primary

and secondary. The secondary source of authentication is used in the event of primary source failover and is optional for configuration. You can configure four possible sources for authentication:

- Local — Use the default device-local database (default)
- RADIUS — Use an external RADIUS server
- LDAP — Use an external LDAP server
- TACACS+ — Use an external TACACS+ server

By default, external AAA services are disabled, and AAA services default to the device-local user database. Any environment requiring more than 64 users should adopt AAA servers for user management.

If the primary source is set to an external AAA service (RADIUS, LDAP, or TACACS+) and the secondary source is not configured, the following events occur:

- For Telnet-based and SSH connections-based logins, the login authentication fails if none of the configured (primary source) AAA servers respond or if an AAA server rejects the login.
- For a serial port (console) connection-based login, if a user's login fails for any reason with the primary source, failover occurs and the same user credentials are used for login through the local source. This failover is not explicit.

Conditions for conformance

Consider the following conditions for remote server authentication:

- If the first source is specified as **default**, do not specify a second source. A second source signals a request to set the login authentication mode to its default value, which is **local**. If the first source is **local**, the second source cannot be set to any value, because the failover will never occur.
- The source of authentication (except **local**) and the corresponding server type configuration are dependent on each other. Therefore, at least one server should be configured before that server type can be specified as a source.
- If the source is configured to be a server type, you cannot delete a server of that type if it is the only server in the list. For example, if there are no entries in the TACACS+ server list, the authentication mode cannot be set to **tacacs+** or **tacacs+ local**. Similarly, when the authentication mode is **radius** or **radius local**, a RADIUS server cannot be deleted if it is the only one in the list.

Configuring remote server authentication

This section introduces the basics of configuring remote server authentication using RADIUS and TACACS+ in a simple manner.

For detailed configuration information on remote server authentication, refer to the following topics:

- Understanding and configuring RADIUS
- Understanding and configuring TACACS+
- Understanding and configuring LDAP

Setting and verifying the login authentication mode

The following procedure configures TACACS+ as the primary source of authentication and the device-local user database as the secondary source. For complete information on login authentication mode, refer to the **aaa authentication login** command in the *Extreme SLX-OS Command Reference*.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

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

2. Enter the **aaa authentication login** command with the specified parameters.

   ```
   device(config)# aaa authentication login tacacs+ local
   ```

3. Enter the **do show running-config aaa** command to display the configuration.

   ```
   device(config)# do show running-config aaa
   aaa authentication login tacacs+ local
   ```

4. Log in to the device using an account with TACACS+-only credentials to verify that TACACS+ is being used to authenticate the user.

Resetting the login authentication mode

The following procedure resets the login configuration mode to the default value using the **no aaa authentication login** command.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

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

2. Enter the **no aaa authentication login** command to remove the configured authentication sequence and to restore the default value (Local only).

   ```
   device(config)# no aaa authentication login
   ```

3. Verify the configuration with the **do show running-config aaa** command.

   ```
   device(config)# do show running-config aaa
   aaa authentication login local
   ```

4. Log in to the device using an account with TACACS+-only credentials. The login should fail with an "access denied" error.

5. Log in to the device using an account with local-only credentials. The login should succeed.

Changing the login authentication mode

You can set the authentication mode with the **aaa authentication login** command.

You can reset the configuration to the default value using the **no aaa authentication login** command.

> **Note**
> In a configuration with primary and secondary sources of authentication, the primary mode cannot be modified alone. First remove the existing configuration and then configure it to the required configuration.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

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

2. Enter the **aaa authentication login** command and specify the desired authentication mode.

   ```
   device(config)# aaa authentication login radius local
   ** or **
   device(config)# aaa authentication login tacacs+ local
   ```

3. Verify the configuration with the **do show running-config aaa** command.

   ```
   device(config)# do show running-config aaa
   aaa authentication login radius local
   ```

4. Log in to the device using an account with TACACS+ credentials. The login should fail with an "access denied" error.

5. Log in to the device using an account with RADIUS credentials. The login should succeed.

RADIUS Server Authentication

RADIUS security

The remote authentication dial-in user service (RADIUS) protocol manages authentication, authorization, and accounting (AAA) services centrally.

You can use a Remote Authentication Dial In User Service (RADIUS) server to secure the following types of access to the Layer-2 device or Layer-3 device:

- Telnet access
- SSH access
- Access to the Privileged EXEC level and CONFIG levels of the CLI, using roles pre-defined on the device and sent as attribute in Radius Response.

RADIUS Authentication

When RADIUS authentication is implemented, the device consults a RADIUS server to verify user names and passwords.

When a device is configured with a set of RADIUS servers to be used for authentication, the device also sends accounting data to the RADIUS server implicitly. When the RADIUS server is not configured to support accounting, the accounting events sent by the device to the server are dropped.

> **Note**
> The device supports the NAS-IPv6-Address attribute (RFC 3162) for RADIUS authentication.

RADIUS Authorization

User authorization through the RADIUS protocol is not supported. The access control of RADIUS users is enforced by the Extreme role-based access control (RBAC) protocol at the device level. A RADIUS user should therefore be assigned a role that is present on the device using the Vendor Specific Attribute (VSA) *Brocade-Auth-Role*. After the successful authentication of the RADIUS user, the role of the user configured on the server is obtained. If the role cannot be obtained or if the obtained role is not present on the device, the user will be assigned the "user" role and a session is granted to the user with "user" authorization.

RADIUS Accounting

Remote Authentication Dial-In User Service (RADIUS) server accounting is supported for recording information about user activity.

RADIUS server accounting supports:
- Login (EXEC) accounting
- Command accounting

> **Note**
> System event accounting is not supported.

When you configure RADIUS accounting on a device, information is sent to the RADIUS accounting server when specific events occur; for example, when a user logs in to the device.

RADIUS accounting works as follows:

1. One of the following events occurs on the device:
   - A user logs in to the management interface using Telnet or SSH.
   - A user enters a command for which RADIUS accounting has been configured.
2. The device checks its configuration to see if the event is one for which RADIUS accounting is required.
3. When the event is concluded, the device sends an accounting stop packet to the RADIUS accounting server.
4. The RADIUS accounting server acknowledges receipt of the accounting stop packet.

> **Note**
> RADIUS server accounting can be enabled and used regardless of whether authentication is performed locally, on a RADIUS server, or on a TACACS+ server. However, RADIUS accounting only takes place after successful authentication.

> **Note**
> In command accounting, commands with a partial timestamp are not accounted.

By default, RADIUS server accounting is disabled. Prior to enabling RADIUS server accounting, at least one RADIUS server must be configured.

For command accounting, the Vendor Specific Attribute (VSA) `Brocade-Cmd` must be added on RADIUS server.

Before downgrading to a software version that does not support RADIUS accounting, both login and command accounting must be disabled.

Account password changes

All existing mechanisms for managing device-local user accounts and passwords remain functional when the device is configured to use RADIUS. Changes made to the device-local database do not propagate to the RADIUS server, nor do the changes affect any account on the RADIUS server; therefore, changes to a RADIUS user password must be done on the RADIUS server.

RADIUS authentication through management interfaces

You can access the device through Telnet or SSH from either the Management interface or the data ports (Ethernet interface or in-band). The device goes through the same RADIUS-based authentication with either access method.

Configuration of an interface as the source of RADIUS packets

You can designate the lowest-numbered IP address configured on Ethernet port, loopback interface, management interface, or virtual interface as the source IP address for RADIUS packets.

When a source interface for RADIUS packets is not configured, the IP address of the interface through which the RADIUS packet exits the device is used as the source IP address in the IP header. Designating a specific interface as the source interface for RADIUS packets provides the following benefits:

- Incoming RADIUS traffic (from each instance of RADIUS server configured on the device), can be directed to particular interfaces using the source interface configuration.
- Firewall configuration is simplified; traffic can be allowed from one well-known source IP address.

When more than one IP address is configured on an interface, the lowest-numbered IP address is used as the source IP address for the RADIUS packets.

A source interface for RADIUS packets must be configured for each instance of RADIUS host that is configured on the device.

You can configure a source interface for RADIUS packets by using the **source-interface** command in RADIUS server host VRF configuration mode.

Configuring server-side RADIUS support

With RADIUS servers, you should set up user accounts by their true network-wide identity, rather than by the account names created on a device. Along with each account name, you must assign appropriate device access roles. A user account can exist on a RADIUS server with the same name as a user on the device at the same time.

When logging in to a device configured with RADIUS, users enter their assigned RADIUS account names and passwords when prompted. Once the RADIUS server authenticates a user, it responds with the assigned device role and information associated with the user account information using an

Extreme Vendor-Specific Attribute (VSA). An Authentication-Accept response without the role assignment automatically grants the "user" role.

> **Note**
> RADIUS Server must be configured to support Vendor-Specific-Attribute (VSA) in addition to configuring RADIUS Server support on the device.

Configuring a RADIUS server with Linux

FreeRADIUS is an open source RADIUS server that runs on all versions of Linux (FreeBSD, NetBSD, and Solaris).

Perform the following steps to configure a RADIUS server with Linux.

1. Download the package from www.freeradius.org and follow the installation instructions at the FreeRADIUS website.
2. Refer to the RADIUS product documentation for information on configuring and starting up a RADIUS server.
3. Determine where vendor-specific dictionaries are located on the server.

```
user@Linux:$ locate dictionary.*

/usr/share/freeradius/dictionary.3com
/usr/share/freeradius/dictionary.3gpp
/usr/share/freeradius/dictionary.3gpp2
/usr/share/freeradius/dictionary.acc
/usr/share/freeradius/dictionary.acme
```

4. Change to the vendor-specific dictionaries directory.

```
user@Linux:$ cd /usr/share/freeradius/
user@Linux:/usr/share/freeradius$
```

5. Verify that the `dictionary.brocade` file exists in this directory.

```
user@Linux:/usr/share/freeradius$ ls dictionary.brocade

dictionary. brocade
```

When the `dictionary.brocade` file does not exist, proceed to Step 7.

6. Check that the contents of the `dictionary.brocade` file are correct. The following example shows the correct information.

```
user@Linux:/usr/share/freeradius$ more dictionary.brocade

# -*- text -*-
# Copyright (C) 2013 The FreeRADIUS Server project and contributors
#
VENDOR          Brocade                         1588
BEGIN-VENDOR    Brocade

ATTRIBUTE       Brocade-Auth-Role                       1       string

END-VENDOR      Brocade
ATTRIBUTE       Brocade-Cmd                             8       string
```

When the `dictionary.brocade` file exists and holds the correct information, proceed to Step 10.

7. When the `dictionary.brocade` file does not exist or holds incorrect information, you need to create a `dictionary.brocade` file with the correct information.

   a. Log in as the root user.

   b. In the vendor-specific dictionaries directory, create a file named `dictionary.brocade` with the below content.

   ```
   # -*- text -*-
   # Copyright (C) 2013 The FreeRADIUS Server project and contributors
   #
   VENDOR          Brocade                              1588
   BEGIN-VENDOR    Brocade

   ATTRIBUTE       Brocade-Auth-Role                       1       string

   END-VENDOR      Brocade
   ```

8. To import the `dictionary.brocade` file, add the following line to the dictionary file.

   ```
   $INCLUDE dictionary.brocade
   ```

9. To ensure that the dictionary is loaded, restart the FreeRADIUS server.

   ```
   user@Linux:/usr/share/freeradius$ sudo service freeradius restart
   ```

10. Configure an Extreme user account.

    a. Open the `/etc/raddb/users` file in a text editor (the location of the FreeRADIUS users configuration file depends on the Linux distribution).

    b. Add the user name and associated the permissions. The user must log in using the permissions specified with Brocade-Auth-Role. The following example configures an account called "jsmith" with admin permissions and a password "jspassword".

    ```
    jsmith    Auth-Type := Local,
                  User-Password == "jspassword",
                  Brocade-Auth-Role = "admin"
    ```

    > **Note**
    > You must use double quotation marks around the password and role.

11. To ensure that the changes take effect, restart the FreeRADIUS server.

    ```
    user@Linux:/usr/share/freeradius$ sudo service freeradius restart
    ```

    > **Note**
    > When you use network information service (NIS) for authentication, the only way to enable authentication with the password file is to force the device to authenticate using password authentication protocol (PAP); this requires the setting the **pap** option with the **radius-server host** command.

Configuring a Windows IAS-based RADIUS server

Step-by-step instructions for installing and configuring Internet Authentication Service (IAS) with Microsoft Windows server 2008 (or earlier versions, Windows 2003 or 2000) can be obtained from www.microsoft.com or your Microsoft documentation. Confer with your system or network administrator prior to configuration for any special needs your network environment may have.

Use the following information to configure the Internet Authentication Service for a device.

> **Note**
> This is not a complete presentation of steps.

1. In the **New RADIUS Client** window, choose **RADIUS Standard** from the **Client-Vendor** menu.
2. Configure the **Dial-in Profile** dialog box as follows:

    a. Select the **Advanced** tab.

    b. Scroll to the bottom of the RADIUS Standard list, select **Vendor-Specific**, and click **Add**.

    The **Multivalued Attribute Information** dialog box appears.

    c. Click **Add** in the **Multivalued Attribute Information** dialog box.

    The **Vendor-Specific Attribute Information** dialog box appears.

    d. Enter the Extreme vendor code value.

    e. Select **Yes. It conforms.** and then click **Configure Attribute**.

    The **Configure VSA (RFC compliant)** dialog box appears.

    f. In the **Configure VSA (RFC compliant)** dialog box, enter the following values and click **OK**:

    - Vendor-assigned attribute number—Enter the value **1**.

    - Attribute format—Enter the value **String**.

    The RADIUS server is now configured.

The following image shows the different screens configured in this task.
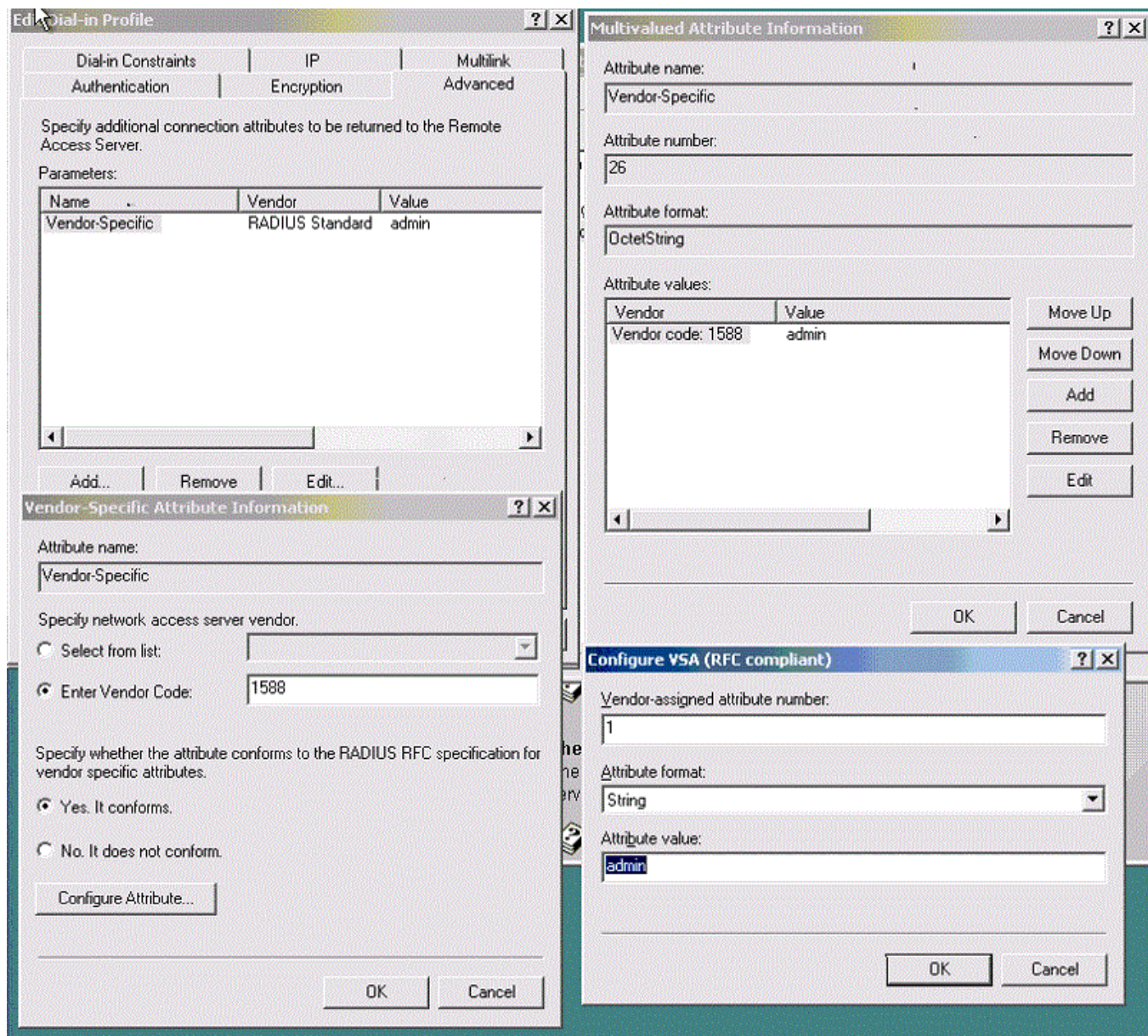
Figure 10: Windows server VSA configuration

Configuring RADIUS Server on a device

Each device client must be individually configured to use RADIUS servers.

You use the **radius-server host** command to specify the server IP address and the VRF through which to communicate with the RADIUS server.

You can configure a maximum of 5 RADIUS servers on a device for AAA service.

> **Note**
> RADIUS Server must be configured to support Vendor-Specific-Attribute (VSA) in addition to configuring RADIUS Server support on the device.

The following table describes configuration commands associated with the VRF used to connect to the RADIUS server.

Table 26: RADIUS server host VRF configuration commands

Command	Description
auth-port	Configures the user datagram protocol (UDP) port used to connect the RADIUS server for authentication. The port range is 0 through 65535; the default port is 1812.
protocol	Configures the authentication protocol to be used. Options include CHAP, PAP, and PEAP. The default protocol is CHAP. IPv6 hosts are not supported if PEAP is the configured protocol.
key	Configures the shared secret between the device and the RADIUS server. The default value is "sharedsecret." The key cannot contain spaces and must be from 8 through 40 characters in length. Empty keys are not supported.
retries	Configures the number of attempts permitted to connect to a RADIUS server. The range is 0 through 100, and the default value is 5.
source-interface	Configures a source IP address for RADIUS packets that originate on the device.
timeout	Configures the time to wait for a server to respond. The range is 1 through 60 seconds. The default value is 5 seconds.
encryption-level	Configures whether the encryption key should be stored in clear-text or in encrypted format. Default is 7 (encrypted). Possible values are 0 or 7, where 0 represents store the key in clear-text format and 7 represents encrypted format.

Note

If you do not configure a shared secret using the **key** command, the authentication session is not encrypted. The shared secret configured using the **key** command must match the value configured in the RADIUS configuration file; otherwise, the communication between the server and the device fails.

There may be situations/configurations where SSH server and RADIUS / TACACS+ server timeouts conflict. The default timeout for the SSH server max-login-timeout is 120 seconds. The default timeout for RADIUS / TACACS+ is 5 seconds, with a retry default of 5 attempts, which may create a scenario where the timeout value is 25 seconds.

Administrators should be aware that the following situation can occur:

If AAA Authentication has been configured with the local-authfallback/local option using five RADIUS / TACACS+ servers; and those servers are not reachable, then the login timeout effectively becomes 125 seconds (25 seconds x 5 servers = 125 seconds). Since the default timeout for the SSH server is 120 seconds, the SSH server will timeout before login can succeed, preventing even the admin from logging in.

It is recommended that Administrators evaluate the default timeouts if this scenario is possible, and make the necessary adjustments to the default values for timeout and retry attempts.

Adding a RADIUS server

You can configure up to five RADIUS servers on a device.

Prior to configuring a RADIUS server by specifying a domain or host name, you must configure the Domain Name System (DNS) server on the device by using the **ip dns** command. The host name cannot be resolved unless the DNS server is configured.

> **Note**
>
> When a list of servers is configured on the device, failover from one server to another server only happens when a RADIUS server fails to respond; it does not happen when user authentication fails.

Perform the following task to add a RADIUS server to a device.

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

```
device# configure terminal

Entering configuration mode terminal
```

2. When the default configuration values for communication with the RADIUS server are not acceptable, use the **radius-server host** command specifying the **use-vrf** parameter to enter RADIUS server host VRF configuration mode.

```
device(config)# radius-server host 10.38.37.180 use-vrf mgmt-vrf
device(config-host-10.38.37.180/mgmt-vrf)#
```

3. The following examples show how to configure some parameters for communication with the RADIUS server using the mgmt-vrf.

   - (Optional) Configure the authentication protocol to use for communication with the RADIUS server.
     ```
     device(config-host-10.38.37.180/mgmt-vrf)# protocol pap
     ```

   - (Optional) Specify a text string to be used as a shared secret between the device and the RADIUS server.
     ```
     device(config-host-10.38.37.180/mgmt-vrf)# key "new#vertigo*secret"
     ```

   - (Optional) Specify the wait time (in seconds) allowed for a RADIUS server response.
     ```
     device(config-host-10.38.37.180/mgmt-vrf)# timeout 10
     ```

   - (Optional) Specify a source interface for RADIUS packets that originate on the device. The following example shows how to configure an Ethernet interface (0/2) as the source interface.
     ```
     device(config-host-10.38.37.180/mgmt-vrf)# source-interface ethernet 0/2
     ```

4. Return to Privileged EXEC mode.

```
device(config-host-10.38.37.180/mgmt-vrf)# end
```

5. Verify the configuration.

```
device# show running-config radius-server host 10.38.37.180

radius-server host 10.38.37.180 use-vrf mgmt-vrf
 protocol pap
 key "ayykN/07wCMEy0SKrpZXPm0hzI37Ze9qNugdSQXhoo0=\n"
 encryption-level 7
 timeout 10
 source-interface ethernet 0/2
```

Importing a RADIUS CA certificate

The following example imports the RADIUS CA certificate from a remote server to a device using secure copy (SCP).

1. In privileged EXEC mode, enter **configure terminal** to change to global configuration mode.

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

2. Enter **crypto ca import radius** with the specified parameters.

   ```
   device# crypto ca import radius directory /usr/radiuscacert file cacert.pem protocol
   SCP host 10.23.24.56 user admin password *****
   ```

3. Verify the import by entering **show crypto ca certificates**.

   ```
   device# show crypto ca certificates
   Trustpoint: t1
   certificate:
   SHA1 Fingerprint=B7:5B:DB:9B:24:69:40:39:36:66:4D:59:2C:69:83:8E:93:CA:23:0C
   Subject: C=US, ST=CA, L=SJ, O=BRC, OU=SFI, CN=10:00:00:27:F8:87:70:29
   Issuer: C=US, ST=CA, L=SJ, O=BR, OU=SF, CN=SOUND/emailAddress=sravi
   Not Before: Oct 6 23:44:27 2014 GMT
   Not After : Oct 6 23:44:27 2015 GMT
   purposes: sslserver
   CA certificate:
   SHA1 Fingerprint=76:5B:D4:2C:CB:54:FE:6B:C5:E0:E3:FD:11:B0:88:70:80:12:C6:63
   Subject: C=US, ST=CA, L=SJ, O=BR, OU=SF, CN=SOUND/emailAddress=sravi
   Issuer: C=US, ST=CA, L=SJ, O=BR, OU=SF, CN=SOUND/emailAddress=sravi
   Not Before: Sep 19 20:56:49 2014 GMT
   Not After : Oct 19 20:56:49 2014 GMT
   purposes: sslserver
   ```

Modifying the RADIUS server configuration

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

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

2. Enter the **radius-server host** command with the help option (**?**) to display the configured RADIUS servers.

   ```
   device(config)# radius-server host ?

   Possible completions:
     INETADDRESS    Domain name or IP Address of this RADIUS server
   ```

3. Enter the **radius-server host** command with the IP address of the server you want to modify and the **use-vrf** option.

   ```
   device(config)# radius-server host 10.38.37.180 use-vrf mgmt-vrf
   ```

   After you run this command you are placed into the RADIUS server host VRF configuration mode where you can specify the parameters you want to modify.

4. Configure the values that you want to change.

   - (Optional) The following example shows how to configure a new key.
     ```
     device(config-host-10.38.37.180/mgmt-vrf)# key "changedsec"
     ```

   - (Optional) The following example shows how configure a timeout value of 3 seconds.
     ```
     device(config-host-10.38.37.180/mgmt-vrf)# timeout 3
     ```

5. Return to Privileged EXEC mode.

```
device(config-host-10.38.37.180/mgmt-vrf)# end
```

6.
Note

This command does not display default values.

Verify the new configuration.

```
device# show running-config radius-server host 10.38.37.180
radius-server host 10.38.37.180 use-vrf mgmt-vrf
 protocol pap key "h8mcoUf2LZF+P+AjaYnO1Q==\n" encryption-level 7 timeout 3
!
```

Note

To remove a server from the list of configured RADIUS servers, use the **no radius-server host** command specifying the IP address or hostname of the RADIUS server that is to be removed.

Configuring the client to use RADIUS for login authentication

After you configured the client-side RADIUS server list, you must set the authentication mode so that RADIUS is used as the primary source of authentication. Refer to the Login authentication mode section for information on how to configure the login authentication mode.

Enabling and disabling login accounting (RADIUS)

Login information can be sent to a Remote Authentication Dial-In User Service (RADIUS) server for accounting purposes.

Before enabling login (EXEC) accounting, at least one RADIUS server host must be configured on the device by using the **radius-server host** command.

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

```
device# configure terminal
```

2. Enable accounting and send login information to a RADIUS accounting server.

```
device(config)# aaa accounting exec default start-stop radius
```

When this command is entered and a RADIUS server is not configured, an error message is displayed to indicate that no active RADIUS server exists to support accounting.

3. Return to privileged EXEC mode.

```
device(config)# exit
device#
```

4. Verify the AAA accounting configuration.

```
device# show running-config aaa accounting

aaa accounting exec default start-stop radius
```

5. (Optional) Once enabled, you can disable sending login information to a RADIUS accounting server.

The following example shows how to disable login accounting by using the **no aaa acccounting** command.

```
device# configure terminal
device(config)# no aaa accounting exec default start-stop
device(config)# exit
```

The following example shows how to disable login accounting by using the **aaa accounting** command specifying the **none** option.

```
device# configure terminal
device(config)# aaa accounting exec default start-stop none
device(config)# exit
```

The following example shows how to enable login accounting on a RADIUS server and verify the configuration.

```
device# configure terminal
device(config)# aaa accounting exec default start-stop radius
device(config)# exit
device# show running-config aaa accounting

aaa accounting exec default start-stop radius
```

Enabling and disabling command accounting (RADIUS)

Command execution information can be sent to a Remote Authentication Dial-In User Service (RADIUS) server for accounting purposes.

Before enabling command accounting, at least one RADIUS server host must be configured on the device by using the **radius-server host** command.

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

```
device# configure terminal
```

2. Enable accounting and send command information to a RADIUS accounting server.

```
device(config)# aaa accounting commands default start-stop radius
```

When this command is entered and a RADIUS server is not configured, an error message is displayed to indicate that no active RADIUS server exists to support accounting.

3. Return to privileged EXEC mode.

```
device(config)# exit
device#
```

4. Verify the AAA accounting configuration.

```
device# show running-config aaa accounting

aaa accounting commands default start-stop radius
```

5. (Optional) Once enabled, you can disable sending command information to a RADIUS accounting server.

The following example shows how to disable command accounting by using the **no aaa acccounting** command.

```
device# configure terminal
device(config)# no aaa accounting commands default start-stop
device(config)# exit
```

The following example shows how to disable command accounting by using the **aaa accounting** command specifying the **none** option.

```
device# configure terminal
device(config)# aaa accounting commands default start-stop none
device(config)# exit
```

The following example shows how to enable command accounting on a RADIUS server and verify the configuration.

```
device# configure terminal
device(config)# aaa accounting commands default start-stop radius
device(config)# exit
device# show running-config aaa accounting

aaa accounting commands default start-stop radius
```

RADIUS two factor authentication support

Traditional password-based authentication methods are based on "one-factor" authentication, where a user confirms an identity using a memorized password. Reliance on one-factor authentication exposes enterprises to increased security risks; passwords may be stolen, guessed, cracked, replayed, or compromised in other ways by unsolicited users by using Man in the Middle Attack.

Two factor authentication increases the security by adding an additional step to the basic log-in procedure which requires the user to have both the password and RSA Secure ID credentials from a hardware token before being able to access a device. Two factor authentication with RADIUS is supported with RSA Manager over PEAP and PAP protocols, and with Google Authenticator over PAP protocol. The authentication proceeds as four basic steps:

First, each hardware token is assigned to a user. It generates an authentication code every 60 seconds using built-in clock and the card's random key (seed). This seed is 128 bits long, is different for each hardware-token, and is loaded into the RSA Secure ID server (RSA Authentication Manager). The token hardware is designed to be tamper-resistant to deter reverse engineering of the token. SLX-OS only supports an RSA ID key fob as a secondary authentication token.

Secondly, the RSA Authentication Manager authenticates the user's password or PIN and token's combination. It takes the clock time as the input value for the encryption process and it is encrypted with the seed record. The resulting value is the token.

Third, the RSA Agent receives authentication requests and forwards them to the RSA Authentication Manager through a secure channel. Based on the response from the Authentication Manager, agents either allow or deny user access.

Finally, the RSA RADIUS Server forwards the user's user ID and passes code to the RSA Authentication Manager, which verifies that the user ID exists and that the pass code is correct for that user at that specific time.

Each RSA Secure ID token holder must have a user record in the RSA Authentication Manager database. The user records must be synchronized in order to operate. These are the options for creating these records:

- Adding data for individual users in the Add User dialog box.
- Copy and edit an existing user record to make a template with group membership and Agent Host activation lists that can be used for many new users.
- Import user data from Security Accounts Manager (SAM) database on a Windows NT system to the Authentication Manager using dumpsamusers.exe and loadsamusers.exe tools.

> **Note**
> RADIUS two factor authentication does not support Challenge Handshake Authentication Protocol (CHAP).

In order to support two factor authentication install RSA Authentication Manager on your Radius Server and set it to accept two-factor authentication input. When the user logs in, the password or token code works automatically without any changes to the device, as shown in the following example.

```
     Welcome to Console Server Management Server

HQ1-4E23-TS1 login: muser34
Password: ****************  <-----For example password/8675309

device#
```

NEW! RADIUS over TLS

The RADIUS protocol is widely deployed client-server model protocol that enables centralized Authentication, Accounting, and Authorization (AAA) over networks.

Transport Layer Security (TLS) is cryptographic protocol to provide communication security between client and server applications that communicate with each other over the network.

The goals of TLS, in order of priority, are as follows:

- Cryptographic security: TLS should be used to establish a secure connection between two parties.
- Interoperability: Independent programmers should be able to develop applications using TLS that can successfully exchange cryptographic parameters without knowledge of another application's code.
- Extensibility: TLS seeks to provide a framework into which new public key and bulk encryption methods can be incorporated as necessary. This framework also accomplishes two sub-goals: preventing the need to create a new protocol (and risking the introduction of possible new weaknesses) and avoiding the need to implement an entire new security library.
- Relative efficiency: Cryptographic operations tend to be highly CPU intensive, particularly public key operations. For this reason, TLS has incorporated an optional session caching scheme to reduce the number of connections that need to be established from scratch. Additionally, care has been taken to reduce network activity.

By default, RADIUS over TCP uses port 2083.

Support for RADIUS over TLS replaces support for RADIUS over UDP. Consider the following as you use RADIUS over TLS:

- Existing logged-in RADIUS over UDP sessions are not terminated when the RADIUS over TLS server is configured.
- The following configuration is not supported: two RADIUS servers with the same IPv4 or IPv6 address, with one server configured with RADIUS over UDP and the other with RADIUS over TLS. The RADIUS host and the VRF are the unique identifiers for each server configuration.
- Fallback to next server or method occurs only when the RADIUS over TLS server is not reachable. For example, if two RADIUS over TLS servers are configured and the first server responds to the authentication request with ACCESS_REJECT, the access to that user is denied without fallback to second server.

Table 27: Related commands

Command	Function
radius-server host	Configures a RADIUS server to connect for external server authentication. The **radsec** option specifies that RADIUS over TLS is to be used.
aaa authentication login	Configures the Authentication, Accounting, and Authorization (AAA) log in sequence. The **radius** option specifies that RADIUS over TLS is to be used.
cipherset radius	Displays the confirmation of Radius cipher list configured successfully message and displays the cipher list.
show cipherset	Displays the configured radius cipher list.

TACACS+ Server Authentication

Understanding and configuring TACACS+

Terminal Access Controller Access-Control System Plus (TACACS+) is an AAA server protocol that uses a centralized authentication server and multiple network access servers or clients. With TACACS+ support, management of devices seamlessly integrates into network fabric environments. After a device is configured to use TACACS+, it becomes a network access server.

TACACS+ authentication, authorization, and accounting

The TACACS+ server is used for authentication, authorization, and accounting. You can access the device through the serial port, or through Telnet or SSH, from either the management interface or the data ports (Ethernet interface or in-band). The device goes through the same TACACS+ authentication with either access method.

Supported TACACS+ packages and protocols

Extreme supports the following TACACS+ packages for running the TACACS+ daemon on remote AAA servers:

- Free TACACS+ daemon. You can download the latest package from www.shrubbery.net/tac_plus.
- ACS 5.3
- ACS 4.2

The TACACS+ protocol v1.78 is used for AAA services between the device client and the TACACS+ server.

The authentication protocols supported for user authentication are Password Authentication Protocol (PAP) and Challenge Handshake Authentication Protocol (CHAP).

TACACS+ configuration components

Configuring TACACS+ requires configuring TACACS+ support on the client (including optional authorization and accounting), as well as configuring TACACS+ on the server. Support for mixed environments might also be required.

Client configuration for TACACS+ support

You must individually configure each device client to use TACACS+ servers. To configure the server IP address, authentication protocols, and other parameters, use the **tacacs-server** command. You can configure a maximum of five TACACS+ servers on a device for AAA service.

The parameters in the following table are associated with a TACACS+ server that is configured on the device.

Table 28: TACACS+ server parameters

Parameter	Description
host	IP address (IPv4 or IPv6) or domain name or host name of the TACACS+ server. Host name requires prior DNS configuration. The maximum supported length for the host name is 40 characters.
port	The TCP port used to connect the TACACS+ server for authentication. The port range is 1 through 65535; the default port is 49.
protocol	The authentication protocol to be used. Options include CHAP and PAP. The default protocol is CHAP.
key	Specifies the text string that is used as the shared secret between the device and the TACACS+ server to make the message exchange secure. The key must be between 1 and 40 characters in length. The default key is **sharedsecret**. The exclamation mark (!) is supported both in RADIUS and TACACS+ servers, and you can specify the password in either double quotes or the escape character (\), for example **"secret!key"** or **secret\!key**. The only other valid characters are alphanumeric characters (such as a-z and 0-9) and underscores. No other special characters are allowed.
retries	The number of attempts permitted to connect to a TACACS+ server. The range is 0 through 100, and the default value is 5.
timeout	The maximum amount of time to wait for a server to respond. Options are from 1 through 60 seconds, and the default value is 5 seconds.
encryption-level	Whether the encryption key should be stored in clear-text or in encrypted format. Possible values are 0 or 7, where 0 represents store the key in clear-text format and 7 represents encrypted format. Default is 7 (encrypted format).
use-vrf	Specifies a VRF though which to communicate with the TACACS+ server.

> **Note**
> If you do not configure the **key** attribute, the authentication session will not be encrypted. The value of **key** must match the value configured in the TACACS+ configuration file; otherwise, the communication between the server and the device fails.

Adding a TACACS+ server to the client server list

Prior to adding the TACACS+ server with a domain name or a host name, you must configure the Domain Name System (DNS) server on the device. Without the DNS server, the TACACS+ server name

resolution fails, which causes the add operation to fail. To configure the DNS server, use the **ip dns** command.

> **Note**
>
> When a list of servers is configured, failover from one server to another server happens only when a TACACS+ server fails to respond; it does not happen when user authentication fails.

The following procedure adds a TACACS+ server host in IPv6 format.

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

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

2. Enter **tacacs-server** and specify the server IP address.

```
device(config)# tacacs-server host fec0:60:69bc:94:211:25ff:fec4:6010 use-vrf mgmt-vrf
```

Upon execution of the command, you are placed into the TACACS server configuration submode where you can specify additional parameters.

3. Specify the additional parameters.

```
device(config)# tacacs-server host fec0:60:69bc:94:211:25ff:fec4:6010
device(config-host-fec0:60:69bc:94:211:25ff:fec4:6010/mgmt-vrf)# protocol chap key
"new#hercules*secret"
device(config-host-fec0:60:69bc:94:211:25ff:fec4:6010/mgmt-vrf)# exit
```

This example specifies the authentication protocol (CHAP).

4. Return to privileged EXEC mode.

```
device(config-tacacs-server-fec0:60:69bc:94:211:25ff:fec4:6010/mgmt-vrf)# end
```

5. Verify the configuration.

```
device# show running-config tacacs-server host fec0:60:69bc:94:211:25ff:fec4:6010
tacacs-server host fec0:60:69bc:94:211:25ff:fec4:6010 use-vrf mgmt-vrf
 key "nPbWil58uf/UJ4UoTUEzGmx/+m8/9fJbHe1uGUH/gM8=\n" encryption-level 7
 !
```

Modifying the client-side TACACS+ server configuration

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

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

2. Display the configured server IP addresses.

```
device(config)# tacacs-server host ?
fec0:60:69bc:94:211:25ff:fec4:6010
```

3. Enter TACACS+ server configuration mode.

```
device(config)# tacacs-server host fec0:60:69bc:94:211:25ff:fec4:6010 use-vrf mgmt-vrf
```

4. Specify the parameters that you want to modify. This example shows how to modify the shared secret key.

```
device(config-tacacs-server-fec0:60:69bc:94:211:25ff:fec4:6010/mgmt-vrf)# key
"changedsec" retries 100
```

5. Return to privileged EXEC mode.

```
device(config-tacacs-server-fec0:60:69bc:94:211:25ff:fec4:6010/mgmt-vrf)# end
```

6. Verify the configuration.

```
device# show running-config tacacs-server host fec0:60:69bc:94:211:25ff:fec4:6010
tacacs-server host fec0:60:69bc:94:211:25ff:fec4:6010 use-vrf mgmt-vrf
 key "h8mcoUf2LZF+P+AjaYnO1Q==\n" encryption-level 7 retries 100
 !
```

This command does not display default values.

Removing the client-side TACACS+ server configuration

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

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

2. Remove a specific TACACS+ server from the list of configured TACACS servers. The following example removes the TACACS+ server fec0:60:69bc:94:211:25ff:fec4:6010.

```
device(config)# no tacacs-server host fec0:60:69bc:94:211:25ff:fec4:6010
```

Device configuration includes a list of configured TACACS+ servers. When authentication, authorization, or accounting mode is set to **tacacs**, you cannot delete the last server in that list. When you attempt to delete the last server on the list, deletion is denied.

Configuring the client to use TACACS+ for login authentication

After you configure the client-side TACACS+ server list, you must set the authentication mode so that TACACS+ is used as the primary source of authentication.

Client configuration for TACACS+ authorization

AAA command authorization is supported for TACACS+

Authorization is the action of determining what a user is allowed to do on a device. By default, TACACS + command authorization is disabled. Regardless of how authentication is performed (whether it is local, or done on a RADIUS or TACACS+ server), when at least one TACACS+ server is configured, you can enable TACACS+ command authorization by using the **aaa authorization command** command.

When TACACS+ command authorization is enabled and a user attempts to run a command, an authorization request is sent to servers on the TACACS+ server list in a round-robin fashion. In response to the authorization request, the TACACS+ server sends either an accept message or a reject message based on the user's configuration or settings on the TACACS+ server. When an accept message is received, the user is permitted to run the command.

At device level, authorization is enforced by role-based control (RBAC). To ensure that local device-level authorization is done when the TACACS+ server is unreachable, enable command authorization by using the **aaa authorization command** command, and specify the **local** option. When the **local** option is not specified, local device-level authorization is not performed when the TACACS+ server is unreachable; therefore, command authorization fails.

TACACS+ command authorization can only be enabled when at least one TACACS+ server is configured. Similarly, when command authorization is enabled, the TACACS+ server cannot be removed when it is the only server on the TACACS+ server list.

Limitations

TACACS+ command authorization:

- Is not supported by REST API or NetConf.
- Is not supported during post boot, or configuration replay.
- If the TACACS+ server is reachable through in-band interface and the local option is not configured for AAA authorization, then the execution of all commands after AAA authorization configuration will fail during a file replay.
- When AAA Authorization is configured and operational REST queries are executed, an Internal Server Error is generated. **Workaround:** Remove the AAA authorization configuration prior to executing operational REST queries.

There may be situations/configurations where SSH server and RADIUS / TACACS+ server timeouts conflict. The default timeout for the SSH server max-login-timeout is 120 seconds. The default timeout for RADIUS / TACACS+ is 5 seconds, with a retry default of 5 attempts, which may create a scenario where the timeout value is 25 seconds.

Administrators should be aware that the following situation can occur:

If AAA Authentication has been configured with the local-authfallback/local option using five RADIUS / TACACS+ servers; and those servers are not reachable, then the login timeout effectively becomes 125 seconds (25 seconds x 5 servers = 125 seconds). Since the default timeout for the SSH server is 120 seconds, the SSH server will timeout before login can succeed, preventing even the admin from logging in.

It is recommended that Administrators evaluate the default timeouts if this scenario is possible, and make the necessary adjustments to the default values for timeout and retry attempts.

Enabling command authorization

Before you enable command authorization, you must configure at least one TACACS+ server by using the **tacacs-server** command. In addition, any TACACS+ server configured for TACACS+ authorization must be configured with user rules (to accept or reject commands).

Perform the following steps to enable TACACS+ command authorization.

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

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

2. Enable command authorization.

```
device(config)# aaa authorization command tacacs+ local
```

This example enables TACACS+ authorization, specifying the **local** option. In the event that the TACACS+ server is unreachable or responds with an error, device-level authorization is performed when the **local** option is specified.

> **Note**
> Supported commands fail when **aaa authorization command** is configured without specifying the **local** option and when the configured TACACS+ servers are not reachable. To recover from this, the "admin" user (only) is allowed to either disable command authorization by using the **aaa authorization command none** command or enable **aaa authorization command** command, specifying the **local** option.

3. Return to privileged EXEC mode.

```
device(config)# exit
```

4. Verify the configuration.

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

aaa authorization tacacs+ local
```

The following example show how to enable and verify TACACS+ command authorization, specifying device-level authorization when the TACACS+ server is unreachable or responds with an error.

```
device# configure terminal
Entering configuration mode terminal
device(config)# aaa authorization command tacacs+ local
device(config)# exit
device(config)# show running-config aaa authorization
aaa authorization tacacs+ local
```

Client configuration for TACACS+ accounting

Once the fundamentals of TACACS+ authentication support are configured on the client, a variety of options are available for tracking user activity.

Client-side TACACS+ accounting overview

The TACACS+ protocol supports accounting as a function that is distinctly separate from authentication. You can use TACACS+ for authentication only, for accounting only, or for both. With a TACACS+ server you can track user logins and the commands that users run during a login session by enabling login accounting, command accounting, or both.

When a TACACS+ server is used for authentication, authorization, or accounting, the device attempts to connect to the first TACACS+ server configured in the list. When the first TACACS+ server cannot be reached, the device attempts to send the packets to the next server on the list.

> **Note**
> When the first server on the list is reachable again, the device sends packets to the first server.

Conditions for conformance

- Only login and command accounting is supported. System event accounting is not supported.
- You can use a TACACS+ server for accounting regardless of whether authentication is performed through RADIUS, TACACS+, or the device-local user database. The only precondition is the presence of one or more TACACS+ servers configured on the device.
- No accounting can be performed if authentication fails.
- In command accounting, commands with a partial timestamp cannot be logged. For example, a **firmware download** command issued with the **reboot** option will not be accounted for, because there is no timestamp available for completion of this command.

Configuring TACACS+ accounting on the client

By default, accounting is disabled on the TACACS+ client (the device), and you must explicitly enable TACACS+. Enabling command accounting and login accounting on the TACACS+ client are two distinct operations. To enable login or command accounting, at least one TACACS+ server must be configured. Similarly, if either login or command accounting is enabled, you cannot remove a TACACS+ server if it is the only server in the list.

Enabling login accounting

The following procedure enables login accounting on a device where accounting is disabled.

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

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

2. Enable login accounting.

```
device(config)# aaa accounting exec default start-stop tacacs+
```

3. Return to privileged EXEC mode.

```
device(config)# exit
```

4. Verify the configuration.

```
device(config)# show running-config aaa accounting
aaa accounting exec default start-stop tacacs+
aaa accounting commands default start-stop tacacs+
```

Enabling command accounting

The following procedure enables command accounting on a device where login accounting is enabled and command accounting is disabled.

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

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

2. Enable command accounting.

```
device(config)# aaa accounting command default start-stop tacacs+
```

3. Return to privileged EXEC mode.

```
device(config)# exit
```

4. Verify the configuration.

```
device# show running-config aaa accounting
aaa accounting exec default start-stop none
aaa accounting commands default start-stop tacacs+
```

Disabling accounting

You have two functionally equivalent options to disable accounting: either by using the **aaa accounting** command with the **none** option or by using the **no aaa accounting** command. You must perform the disable operation separately for login accounting and for command accounting. The operation is performed in global configuration mode.

The following example shows how to disable command accounting by using the **aaa accounting** command with the **none** option.

```
device(config)# aaa accounting commands default start-stop none
```

The following example shows how to disable command accounting by using the **no aaa accounting** command.

```
device(config)# no aaa accounting commands default start-stop
```

The following example shows how to disable login accounting by using the **aaa accounting** command with the **none** option.

```
device(config)# aaa accounting exec default start-stop none
```

The following example shows how to disable login accounting by using the **no aaa accounting** command.

```
device(config)# no aaa accounting exec default start-stop
```

Viewing the TACACS+ accounting logs

The following excerpts from TACACS+ accounting logs exemplify typical success and failure cases for command accounting and login accounting.

The following examples were taken from the free TACACS+ server. The order of the attributes might vary depending on the server package, but the values are the same. The location of the accounting logs depends on the server configuration.

Command accounting examples

The following example shows a successful execution of the **shutdown** command by the admin user, followed by a **no shutdown** command.

```
Wed Oct 14 10:40:40 2015          10.18.245.157   admin1  /dev/pts/0      10.70.7.36
stop    task_id=1       timezone=Etc/GMT        service=shell   priv-lvl=0
Cmd="operational top configure terminal" Stop_time=Wed Oct 14 17:39:49 2015

        Status=Succeeded

Wed Oct 14 10:42:14 2015          10.18.245.157   admin1  /dev/pts/0      10.70.7.36
stop    task_id=1       timezone=Etc/GMT        service=shell   priv-lvl=0
Cmd="configure conf-if-eth-0/3 shutdown"       Stop_time=Wed Oct 14 17:41:24 2015

        Status=Succeeded

Wed Oct 14 10:42:23 2015          10.18.245.157   admin1  /dev/pts/0      10.70.7.36
stop    task_id=1       timezone=Etc/GMT        service=shell   priv-lvl=0
Cmd="configure conf-if-eth-0/3 no shutdown"    Stop_time=Wed Oct 14 17:41:33 2015
```

The following example shows a successful execution of the **username** command by the admin user.

```
<102> 2012-04-09 15:21:43 4/9/2012 3:21:43 PM NAS_IP=10.17.37.150 Port=0 rem_addr=Console
User=admin Flags=Stop task_id=1 timezone=Etc/GMT+0 service=shell priv-lvl=0 Cmd=username
Stop_time=Mon Apr 9 09:43:56 2012
 Status=Succeeded
```

The following example shows a failed execution of the **radius-server** command by the admin user due to an invalid host name or server IP address.

```
Aug 19 20:57:12 10.24.12.77      admin   /dev/pts/0      10.252.200.38   stop
task_id=1       timezone=Etc/ config radius-server host 10.2.3"      Stop_time=Fri Aug 19
08:25:56 2016
        Status=%% Error: Invalid host name or IP address
```

Login (EXEC) accounting examples

The following example shows a successful login of the trial user.

```
Aug 19 21:01:46 10.24.12.77      user   /dev/pts/1      10.252.200.38   start
task_id=1       timezone=Etc/GMT     service=shell
```

The following example shows a successful logout of the trial user.

```
Aug 19 21:03:11 10.24.12.77      user   /dev/pts/1      10.252.200.38   stop
task_id=1       timezone=Etc/GMT     service=shell   elapsed_time=85 reason=admin reset
```

Configuring TACACS+ on the server side

Step-by-step instructions for installing and configuring can be obtained from your server manufacturer. Confer with your system or network administrator prior to configuration for any special needs your network environment might have.

Server-side user account administration overview

With TACACS+ servers, you should set up user accounts by their true network-wide identity, rather than by the account names created on a device. Along with each account name, you must assign appropriate device access roles. A user account can exist on TACACS+ servers with the same name as a user on the device at the same time.

When logging in to a device configured with a TACACS+ server, users enter their assigned TACACS+ account names and passwords when prompted. After the TACACS+ server authenticates a user, it responds with the assigned device role and user account information, using an Extreme Vendor-Specific Attribute (VSA). An Authentication-Accept response without the role assignment automatically grants the "user" role.

User accounts, protocols passwords, and related settings are configured by editing the server configuration files.

Establishing a server-side user account

The following example assigns the user "Mary" the Extreme role of "vlanadmin" and different passwords depending on whether CHAP or PAP is used. In the following example, which works in an environment with only devices supported by this guide, the brcd-role attribute is mandatory. In a mixed-vendor

environment, the brcd-role attribute must be set to optional. Refer to Configuring TACACS+ for a mixed-vendor environment on page 138 for more information.

```
user = Mary {
chap = cleartext "chap password"
pap = cleartext "pap password"
service = exec {
brcd-role = vlanadmin;
}
}
```

The following example assigns the user "Agnes" a single password for all types of login authentication.

```
user = Agnes {
global = cleartext "Agnes global password"
}
```

Alternatively, a user can be authenticated using the /etc/passwd file. The following example allows the user "fred" to be authenticated using the /etc/passwd file.

```
user = fred {
login = file /etc/passwd
}
```

Changing a server-side TACACS+ account password

To change a TACACS+ user password, edit the configuration on the TACACS+ server.

Defining a server-side TACACS+ group

A TACACS+ group or role can contain the same attributes as user accounts. By inference, all the attributes of a group can be assigned to any user to whom the group is assigned. The TACACS+ group, while functionally similar to the Extreme role concept, has no relation with the value of the "brcd-role" attribute.

The following example defines a TACACS+ group.

```
group = admin {
# group admin has a cleartext password which all members share
# unless they have their own password defined
chap = cleartext "my$parent$chap$password"
}
```

The following example assigns the user "Extreme" with the group "admin".

```
user = Extreme {
member = admin
pap = cleartext "pap password"
}
```

Setting a server-side account expiration date

You can set an expiration date for an account by using the "expires" attribute in the TACACS+ server configuration file. The expiration date has the format *"MMM DD YYYY"*.

```
user = Extreme {
member = admin
expires = "Jan 01 2017"
pap = cleartext "pap password"
}
```

Configuring a TACACS+ server key

The TACACS+ server key is the shared secret used to secure the messages exchanged between the device and the TACACS+ server. The TACACS+ server key must be configured on both the TACACS+ server and the client device. Only one key is defined per server in the TACACS+ server configuration file. The key is defined as follows:

```
key = "shared secret text"
```

Configuring TACACS+ for the AAA user role

Configuring TACACS+ for the AAA user role allows the AAA user role to access configuration commands.

At least one TACACS+ server must be configured on the device using the **tacacs-server host** command.

You must configure a server-side user role on the TACACS+ server. Refer to Configuring TACACS+ for a mixed-vendor environment on page 138 for more information.

The following example assigns the user "Agnes" a single password for all types of login authentication.

```
user = Agnes {
global = cleartext "Agnes global password"
}
```

Configuring server-side rules for TACACS+ command authorization

To perform TACACS+ command authorization, you must configure a TACACS+ server with user rules to accept or reject commands.

The following example shows a rule configuration for a user named **tacuser**. In this configuration, a reject message is returned for the **show vrf** command and an accept message is returned for all other **show** commands.

```
user = tacuser {
          default service = permit
          chap = cleartext "password"
          service = exec {
            brcd-role = admin
          }
          cmd = show {
           deny vrf
               permit .*
          }
  }
```

Configuring TACACS+ for a mixed-vendor environment

Extreme uses Role-Based Access Control (RBAC) to authorize access to system objects by authenticated users. In AAA environments, users might need to be authorized across platforms supported by this guide and other platforms. You can use TACACS+ to provide centralized AAA services to multiple network access servers or clients. To use TACACS+ services in mixed-vendor environments, you must configure the Attribute-Value Pair (AVP) argument to be optional, as shown in the following example.

```
brcd-role*admin
```

The device sends the optional argument **brcd-role** in the authorization request to the TACACS+ server. Most TACACS+ servers are programmed to return the same argument in response to the authorization request, If "brcd-role" is configured as an optional argument, it is sent in the authorization request, and Extreme users are able to successfully authorize with all TACACS+ servers in a mixed-vendor environment.

Configuring optional arguments in tac_plus

The following example is specific to the tac_plus package. The syntax for other packages might differ.

In the example, the mandatory attribute priv-lvl=15 is set to allow the server to authenticate. The optional brcd-role = admin argument is added to the tac_plus.conf file and allows devices to authenticate.

The following example configures a user with the optional AVP, brcd-role = admin. An Extreme user must match both the *username* and *usergroup* to authenticate successfully.

```
user = <username> {
    default service = permit
    service = exec {
     priv-lvl=15
  optional brcd-role = admin
  }
}
```

or

```
group = <usergroup> {
    default service = permit
    service = exec {
     priv-lvl=15
     optional brcd-role = admin
  }
}
user = <username> {
   Member = <usergroup>
}
```

Commands not supported for TACACS+ accounting

The following tables list commands not supported for TACACS+ accounting.

Table 29: Privileged EXEC mode commands not supported for TACACS+ accounting

Command name	Command Description
cipherset	Configures FIPS-compliant secure ciphers for LDAP and SSH.
copy	Copies data.
delete	Delete a specified file.
dir	Displays a directory listing.
exit	Exits to the top level and optionally runs a command.
fips	Executes FIPS-related operations.
firmwaredownload	Downloads firmware.
help	Provides help information.

Table 29: Privileged EXEC mode commands not supported for TACACS+ accounting (continued)

Command name	Command Description
history	Configures the size of the history log.
logout	Terminates the current login session.
ping	Executes the **ping** command.
quit	Terminates the current session.
rename	Renames a file.
reload	Reboots the system.
show cipherset	Displays ciphers for LDAP and SSH.
show cli	Displays CLI session parameters.
show file	Displays the contents of a file.
show history	Displays command history.
show netconf-state	Displays NETCONF statistics.
show parser dump	Displays a parser dump.
show running-config	Displays the running configuration.
show startup-db	Displays the startup configuration.
show startup-config	Displays the contents of the startup-configuration file.
terminal	Configures terminal properties.
traceroute	Executes the **traceroute** command.

Table 30: Global configuration mode commands not supported for TACACS+ accounting

Command name	Command Description
abort	Aborts the current configuration session.
end	Terminates the current configuration session.
exit	Exits from the current mode.
help	Provides help information.
service	Performs password encryption services.
top	Exits to the top level and optionally runs a command.

Key Chain Authentication

NEW! Key Chain Authentication Overview

Key chain authentication is the process of ensuring that the key of "person A" held by "person B" belongs to "person A" and vice versa.

Key authentication is used to solve the problem of authenticating the keys of the person (say "person B") to whom some other person ("person A") is talking to or trying to talk to. A symmetric key scheme is supported for authentication.

A key-authenticated agreement method is one in which two or more parties establish cryptographic keys based on one or more party's knowledge of a password. It supports SHA-1, SHA-256, SHA-384, and SHA-512 keyed hash algorithms. The digest is calculated by prepending the actual secret key to the packet header and hashed by one of the supported algorithms. The key ID and the calculated digest form the Message Authentication Code (MAC).
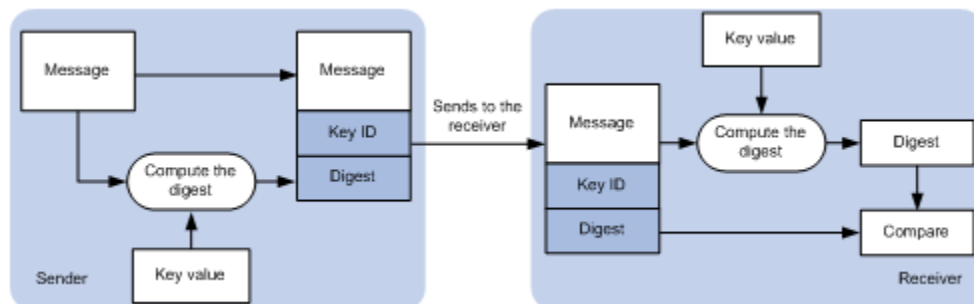

Figure 11: Key chain authentication

With this feature, routing protocols such as BGP4, IS-IS, OSPF, and OSPFv3 use the global authentication key chain configuration for hitless key rollover. Authentication keys can be configured as

key chains. Key chains are sequences of keys (shared secrets). You can use key-based authentication to secure communications with other devices and you can periodically rotate the keys in the chain.

Consider the following when you use global authentication key chains.

- A key chain is a sequence of keys that are collectively managed for authenticating peer.
- Under `authentication key-chain` mode, you can configure a series of key IDs and associate the lifetime and hash algorithm options (SHA1, SHA256, SHA384, or SHA512).
- You can configure a maximum of 128 key chains, with a maximum of 8 key per key chain.

NEW! Configure a Key Chain

You can create no more than 128 key chains.

A key chain name is an alphanumeric string, with a minimum of 4 characters and a maximum of 32 characters.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure a key chain.

```
device(config)# keychain keychain1
```

This example configures a key chain named keychain1.

3. Repeat step 2 for each key chain that you need.

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# keychain keychain1
```

NEW! Configure a Key Accept Tolerance

Accept tolerance is the number of seconds for which expired or soon-to-be activated keys can be used for validating received packets.

You can use this command to extend the validity of an expired key to ensure a smooth key rollover for the processing of a received packet. You can use this command to decrease the activation time of a new key so that a received packet can be authenticated with the new key. A longer accept tolerance period can reduce security if an old key was exposed.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter keychain configuration mode.

```
device(config)# keychain keychain1
```

This example enters configuration mode for key chain 1.

3. Configure the accept tolerance.

```
device(config-keychain1)# accept-tolerance 500
```

This example configures an accept tolerance of 500 seconds in key chain 1. The default is 600 seconds. Valid values range from 0 to 600.

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# keychain keychain1
device(config-keychain1)# accept-tolerance 500
```

NEW! Configure a Key ID

You can configure a unique key ID in a key chain and enter key configuration mode

You can configure no more than 8 keys per key chain.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter keychain configuration mode.

   ```
   device(config)# keychain keychain1
   ```

   This example enters configuration mode for key chain 1.

3. Configure the key.

   ```
   device(config-keychain1)# key 10
   ```

   This example configures a key ID of 10 in key chain 1. Valid ID values range from 1 to 65535.

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# keychain keychain1
device(config-keychain1)# key 10
device(config-keychain1-key10)#
```

NEW! Configure a Key Lifetime

You can define the time period when a key is active.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter keychain configuration mode.

   ```
   device(config)# keychain keychain1
   ```

   This example enters configuration mode for key chain 1.

3. Enter key configuration mode.

   ```
   device(config-keychain1)# key 10
   ```

   This example enters configuration mode for key 10 in key chain 1.

4. Specify the key starting time and day and the ending time and day.

   ```
   device(config-keychain1-key10)# accept-lifetime 00:00:00|06/04/2020 23:59:59|12/04/2020
   ```

   This example configures a lifetime from June 6 2020 to December 4 2020 (UTC) for key 10 in key chain 1.

This example configures a lifetime from June 6 2020 to December 4 2020 (local) for key 10 in key chain 1.

```
device# configure terminal
device(config)# keychain keychain1
device(config-keychain1)# key 10
```

```
device(config-keychain1-key10)# accept-lifetime local true 00:00:00|06/04/2020 23:59:59|
12/04/2020
```

This example configures a lifetime that never expires for key 10 in key chain 1.

```
device# configure terminal
device(config)# keychain keychain1
device(config-keychain1)# key 10
device(config-keychain1-key10)# accept-lifetime infinite
```

NEW! Configure a Key Algorithm

You can define the hash algorithm type for a specified key.

You can choose from one of the following algorithms: HMAC SHA-1, HMAC SHA-256, HMAC SHA-384, and HMAC SHA-512

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter keychain configuration mode.

   ```
   device(config)# keychain keychain1
   ```

   This example enters configuration mode for key chain 1.

3. Enter key configuration mode.

   ```
   device(config-keychain1)# key 10
   ```

   This example enters configuration mode for key 10 in key chain 1.

4. Specify the algorithm.

   ```
   device(config-keychain1-key10)# key-algorithm HMAC-SHA-256
   ```

   This example configures SHA-256 for key 10 in key chain 1.

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# keychain keychain1
device(config-keychain1)# key 10
device(config-keychain1-key10)# key-algorithm HMAC-SHA-256
```

NEW! Display Key Chain Configuration Details

You can display the details of a configured key chain.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Display key chain details.

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

   This example shows details for a key chain named "child."

```
keychain child
 accept-tolerance 500
 key 1
  key-string $9$XutLBELmbQ765dsLycIP/A== encryption-level 4
  accept-lifetime local true 11:49:11|11/09/2017 11:45:16|11/10/2017
```

```
  key-algorithm HMAC-SHA-256
!
```

Lightweight Directory Access Protocol

Understanding and configuring LDAP

Lightweight Directory Access Protocol (LDAP) is an open-source protocol for accessing distributed directory services that act in accordance with X.500 data and service models. LDAP assumes that one or more servers jointly provide access to a Directory Information Tree (DIT) where data is stored and organized as entries in a hierarchical fashion. Each entry has a name called the distinguished name that uniquely identifies it.

LDAP can also be used for centralized authentication through directory service.

Active Directory (AD) is a directory service that supports a number of standardized protocols such as LDAP, Kerberos authentication, and Domain Name Server (DNS), to provide various network services. AD uses a structured data store as the basis for a logical, hierarchical organization of directory information. AD includes user profiles and groups as part of directory information, so it can be used as a centralized database for authenticating third-party resources.

User authentication

A device can be configured as an LDAP client for authentication with an Active Directory (AD) server, supporting authentication with a clear text password over the Transport Layer Security (TLS) channel. Optionally, the device supports server authentication during the TLS handshake. Only the user principal name from the AD server is supported for LDAP authentication on the device. The common name (CN) based authentication is not supported. When you log in from the device, the complete user principal name, including domain, should be entered (for example, "testuser@sec.example.com").

LDAP supports alternative user principal names, such as:

- username
- username@AD.com

- username@ADsuffix.com
- username@newUPN.com

A device configured to perform LDAP-based authentication supports access through a serial port, Telnet, and SSH. These access channels require that you know the device IP address or name to connect to the device.

A maximum of five AD servers can be configured on a device.

Server authentication

As a part of user authentication using LDAP, the device can be configured to support server certificate authentication. To enable server authentication (server certificate verification), follow these guidelines:

- While configuring the LDAP server, the Fully Qualified Domain Name (FQDN) of the AD server must be added as the host parameter, instead of the IP address. An FQDN is needed to validate the server identity as mentioned in the common name of the server certificate.
- The CA certificate of the AD server's certificate must be installed on the device. Currently, only PEM-formatted CA certificates can be imported into the device.

If more than one server is configured and an LDAP CA certificate is imported for one server on the device, the device performs the server certificate verification on all servers. Thus, either CA certificates for all servers must be imported, or CA certificates must not be imported for any of the servers. After the CA certificate is imported, it is retained even if the device is set back to its default configuration. If the CA certificate is not required, you must explicitly delete it.

> **Note**
> The LDAP CA certificate is mandatory for the LDAPS (LDAP over TLS) mode of operation.

Server authorization

The Active Directory (AD) server is used only for authentication. Command authorization of the AD users is not supported in the AD server. Instead, the access control of AD users is enforced locally by role-based access control (RBAC) on the device.

A user on an AD server must be assigned a nonprimary group, and that group name must be either matched or mapped to one of the existing roles on the device; otherwise, authentication will fail. After successful authentication, the device receives the nonprimary group of the user from the AD server and finds the corresponding user role for the group based on the matched or mapped roles.

If the device fails to get the group from the AD server, or the LDAP user is not a member of any matching AD group, the user authentication fails. Groups that match with the existing device roles have higher priority than the groups that are mapped with the device roles. Thereafter, the role obtained from the AD server (or default role) is used for RBAC.

If multiple nonprimary groups are associated to the AD user, only one of the groups must be mapped or matched to the device role. If multiple AD groups of AD users are mapped or matched to the device roles, authentication of the user is successful, but there is no guarantee as to which role the AD user gets among those multiple roles. After successful authentication, the device gets the nonprimary group of the user from the AD server and finds the corresponding user role for the group based on the

matched or mapped roles. Thereafter, the role obtained from the AD server (or default role) will be used for RBAC.

A maximum of 16 AD groups can be mapped to the device roles.

FIPS compliance

To support FIPS compliance, the CA certificate of the AD server's certificate must be installed on the device, and the FIPS-compliant TLS ciphers for LDAP must be used.

Configuring LDAP

Configuring support for LDAP requires configuring both the client and the server. This following major tasks are sorted by client-side and server-side activities:

Client-side tasks:

- Configuring an Active Directory server on the client side on page 150
- Configuring Active Directory groups on the client side on page 151

Server-side tasks:

- Creating a user account on an LDAP/AD server on page 152
- Verifying the user account on a device on page 153
- Configuring LDAP users on a Windows AD server on page 153

Importing an LDAP CA certificate

The following example imports the LDAP CA certificate from a remote server to a device using secure copy (SCP).

1. In privileged EXEC mode, enter **configure terminal** to change to global configuration mode.

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

2. Enter **crypto ca import ldapca** with the specified parameters.

```
device# crypto ca import ldapca directory /usr/ldapcacert file cacert.pem protocol SCP
host 10.23.24.56 user admin password *****
```

3. Verify the import by entering **show crypto ca certificates**.

```
device# show crypto ca certificates
Trustpoint: t1
certificate:
SHA1 Fingerprint=B7:5B:DB:9B:24:69:40:39:36:66:4D:59:2C:69:83:8E:93:CA:23:0C
Subject: C=US, ST=CA, L=SJ, O=BRC, OU=SFI, CN=10:00:00:27:F8:87:70:29
Issuer: C=US, ST=CA, L=SJ, O=BR, OU=SF, CN=SOUND/emailAddress=sravi
Not Before: Oct 6 23:44:27 2014 GMT
Not After : Oct 6 23:44:27 2015 GMT
purposes: sslserver
CA certificate:
SHA1 Fingerprint=76:5B:D4:2C:CB:54:FE:6B:C5:E0:E3:FD:11:B0:88:70:80:12:C6:63
Subject: C=US, ST=CA, L=SJ, O=BR, OU=SF, CN=SOUND/emailAddress=sravi
Issuer: C=US, ST=CA, L=SJ, O=BR, OU=SF, CN=SOUND/emailAddress=sravi
Not Before: Sep 19 20:56:49 2014 GMT
Not After : Oct 19 20:56:49 2014 GMT
purposes: sslserver
```

Viewing the LDAP CA certificate

The following procedure allows you to view the LDAP CA certificate that has been imported on the device.

1. Connect to the device and log in using an account with admin role permissions.
2. In privileged EXEC mode, enter the **show crypto ca certificates** command.

```
device# show crypto ca certificates
```

Configuring an Active Directory server on the client side

Each device client must be individually configured to use Active Directory servers. You can configure a maximum of five Active Directory servers on a device for AAA service.

The parameters in the following table are associated with an Active Directory server that is configured on the device.

Table 31: Active Directory parameters

Parameter	Description
host	IPv4 or Fully Qualified Domain Name of the AD server. IPv6 is supported for Windows 2008 AD server only. The maximum supported length for the host name is 40 characters.
port	TCP port used to connect the AD server for authentication. The valid port range is 1024 through 65535. The default port is 389.
timeout	Time to wait for a server to respond. The range is 1 through 60 seconds. The default value is 5 seconds.
retries	Number of unsuccessful attempts to be made to connect to an AD server before quitting. The valid range is 1 through 100. The default value is 5.
domain	Base domain name.

Adding an LDAP server to the client server list

The following procedure configures an LDAP server on an LDAP client device.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

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

2. Use the **ldap-server-host** command to set the parameters for the LDAP server.

   This command places you into the LDAP server configuration submode where you can modify the server default settings.

```
device(config)# ldap-server host 10.24.65.6
device(config-host-10.24.65.6/mgmt-vrf)#
```

3. Modify any settings, such as the domain name or retry limit, in this configuration mode (refer to the table in Configuring an Active Directory server on the client side on page 150).

```
device(config-host-10.24.65.6/mgmt-vrf)# basedn security.brocade.com
device(config-host-10.24.65.6/mgmt-vrf)# port 3890 timeout 8
device(config-host-10.24.65.6/mgmt-vrf)# retries 3
```

4. Confirm the LDAP settings with the **do show running-config ldap-server** command.

   Attributes holding default values are not displayed.

   ```
   device(config-host-10.24.65.6/mgmt-vrf)# do show running-config ldap-server host
   10.24.65.6
   ldap-server host 10.24.65.6 use-vrf mgmt-vrf
   port 3890 retries 3 timeout 8 basedn security.brocade.com
   ```

5. Use the **exit** command to return to global configuration mode.

   ```
   device(config-host-10.24.65.6/mgmt-vrf)# exit
   ```

Changing LDAP server parameters

Changing the LDAP server parameters follows the same procedure as that noted for adding an LDAP server to the client server list. Enter the host IP address or host name, and then enter the new values as required.

Refer to Adding an LDAP server to the client server list on page 150.

```
device# configure terminal
Entering configuration mode terminal
device(config)# ldap-server host 10.24.65.6
device(config-host-10.24.65.6/mgmt-vrf)# basedn security.brocade.com
```

Removing an LDAP server

The following procedure deletes an LDAP server entry from the device LDAP server list.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode

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

2. Use the **no ldap-server** command to delete the LDAP server.

   ```
   device(config)# no ldap-server host 10.24.65.6
   ```

Configuring Active Directory groups on the client side

An Active Directory (AD) group defines access permissions for the LDAP server similar to Extreme roles. You can map an Active Directory group to an Extreme role with the **ldap-server maprole** command. The command confers all access privileges defined by the Active Directory group to the Extreme role to which it is mapped.

A user on an AD server must be assigned a nonprimary group, and that group name must be either matched or mapped to one of the existing roles on the device.

After successful authentication, the user is assigned a role from a nonprimary group (defined on the AD server) based on the matched or mapped device role.

A user logging in to the device that is configured to use LDAP and has a valid LDAP user name and password will be assigned LDAP user privileges if the user is not assigned a role from any nonprimary group.

Mapping an Active Directory group to a device role

In the following example, a user with the admin role inherits all privileges associated with the Active Directory (AD) Administrator group.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

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

2. Use the **ldap-server maprole** command to set the group information.

A maximum of 16 AD groups can be mapped to the device roles.

```
device(config)# ldap-server maprole group Administrator role admin
```

Removing the mapping of an Active Directory to a device role

The following example removes the mapping between the Extreme admin role and the Active Directory (AD) Administrator group. A user with the admin role can no longer perform the operations associated with the AD Administrator group.

To unmap an AD group to a device role, perform the following steps.

1. In privileged EXEC mode, use the **configure terminal** command to enter global configuration mode.

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

2. Use the **no ldap-server maprole** command to set the group information.

```
device(config)# no ldap-server maprole group Administrator
```

Configuring the client to use LDAP/AD for login authentication

After you configure the device LDAP server list, you must set the authentication mode so that LDAP is used as the primary source of authentication.

Refer to Login authentication mode on page 109 for information on how to configure the login authentication mode.

Configuring an Active Directory server on the server side

The following high-level overview of server-side configuration for LDAP/AD servers indicates the steps needed to set up a user account. This overview is provided for your convenience only. All instructions involving Microsoft Active Directory can be obtained from www.microsoft.com or from your Microsoft documentation. Confer with your system or network administrator prior to configuration for any special needs your network environment may have.

Creating a user account on an LDAP/AD server

The following procedure configures a user account on an LDAP/AD server.

1. Create a user on the Microsoft Active Directory server.

2. Create a group. The group should match with the user's Extremedevice role.

3. Optional: You can map the role to the Extreme device role with the **ldap-server maprole** command.

4. Associate the user with the group by adding the user to the group.

   The user account configuration is complete.

Verifying the user account on a device

The following procedure verifies a user account on a device.

1. Log in to the device as a user with admin privileges.

2. Verify that the LDAP/AD server has an entry in the device LDAP server list.

   ```
   device# show running-config ldap-server
   ```

3. In global configuration mode, set the login authentication mode on the device to use LDAP only and verify the change.

   ```
   device# configure terminal
   Entering configuration mode terminal
   device(config)# no aaa authentication login
   device(config)# aaa authentication login ldap
   device(config)# do
    show running-config aaa
   aaa authentication login ldap
   ```

4. Log in to the device using an account with valid LDAP/AD only credentials to verify that LDAP/AD is being used to authenticate the user.

5. Log in to the device using an account with device-local only credentials. The login should fail with an access denied message.

Configuring LDAP users on a Windows AD server

The following procedure configures a user account on a Windows AD server.

1. Create a user in Windows.

   a. Open **Programs** > **Administrative Tools** > **Active directory Users and Computers**.

   b. Add a user by completing the **Active directory Users and Computers** dialog box.

   c. Save the account information.

   d. From a command prompt, log in using the new user name and enter a password when prompted.

2. Create a group in Windows.

   a. Go to **Programs** > **Administrative Tools** > **Active directory Users and Computers**.

   b. Add a new group.

   c. Save the group information.

3. Assign the group to the user.

   a. Click on the user name.

   b. From the **Properties** dialog box, click the **Member Of** tab and update the field with the group name. This group should either match the device role or it must be mapped with the device role on the device. In this instance, Domain Users is the primary group and therefore should not be mapped with the device role.

NEW! LDAP over TLS

Lightweight Directory Access Protocol (LDAP) is used in Authentication, Accounting, and Authorization (AAA) server environments that consist of a centralized authentication server and multiple Network Access Servers (NAS) or clients. With LDAP support, management of SLX devices integrates seamlessly into these environments.

Transport Layer Security (TLS) is cryptographic protocol to provide communication security between client and server applications that communicate with each other over the network.

The certificate revocation status of the LDAP over TLS (LDAPS) client can be checked using OCSP. LDAPS protects communication when it is established.

By default, LDAPS uses port 636.

Support for LDAPS replaces support for startTLS mode. Consider the following as you use LDAPS:

- Existing logged-in startTLS sessions are not terminated when the LDAPS server is configured.
- The following configuration is not supported: two LDAP servers with the same IPv4 or IPv6 address (and the same VRF), with one server configured with startTLS and the other with LDAPS. The LDAP host and the VRF are the unique identifiers for each server configuration.
- Unlike startTLS, LDAPS authentication fails without an imported CA certificate. For more information, see the **crypto import** command in the *Extreme SLX-OS Command Reference*.

Table 32: Related commands

Command	Function
ldap-server host	Configures an LDAP server to connect for external or remote authentication. The **ldaps** option specifies that LDAP over TLS is to be used.
ldap-server source-interface	Configures the LDAP server on specific VRF with source interface.
crypto import	Imports the Identity Certificate for security configuration. The **ldapca** option specifies that LDAP over TLS is to be used.
cipherset ldap	Displays the confirmation of LDAP cipher list configured successfully message and displays the cipher list.
show cipherset	Displays the configured LDAP cipher list.

OAuth2 Authentication

NEW! OAuth2 Authentication

Support for OAuth2 authentication allows the processing of authentication requests from north-bound interfaces using an OAuth2 token.

Overview

- OAuth2 Authentication is supported only for SLX 9250.
- The SLX device uses a PKI certificate to validate the incoming token.
- Use the **crypto import** command with the **oauth2pkicert** option to import the OAuth2 PKI certificate.
- Use the **aaa authentication login oauth2 [local | local-auth-fallback]** command to configure the OAuth2 mode of authentication.
- OAuth2 tokens are supported by the Bearer Token field in the HTTPS authorization header.
- The **user** command in the audit log shows the xpath format for NETCONF configurations. For example:

```
SLX# show logging auditlog
0 AUDIT, 2020/02/20-22:45:57 (GMT), [DCM-1006], INFO, DCMCFG, admin/admin/
134.141.219.78/ssh/netconf,, SLX, Event: database commit transaction, Status:
Succeeded, User command:  /brocade-interface:interface/ethernet[name="0/26"]/
switchport-basic/basic.
1 AUDIT, 2020/02/20-22:52:23 (GMT), [DCM-1018], WARNING, DCMCFG, admin/admin/
134.141.219.78/ssh/netconf,, SLX, Event: database commit transaction, Status: %%
Error: Remove L3 configuration from the interface, User command: /brocade-
interface:interface/ethernet[name="0/1"]/switchport-basic/basic, /brocade-
interface:interface/ethernet[name="0/2"]/switchport-basic/basic, /brocade-
interface:inte.
```

Considerations

- Setting the AAA authentication mode to OAuth2 is applicable to the SSH (NETCONF) and RESTCONF login methods. Telnet login using the OAuth2 token always fails because it is a non-

secure means of transferring the OAuth2 token. As a best practice, set the secondary source of authentication in the `aaa authentication` command to always fall back to local authentication.

- Unlike other remote server authentication mode of operations, OAuth2 with `local` or `local-auth-fallback` always falls back to the local mode of authentication if the primary source fails.
- By default, any role from the OAuth2 token is mapped to the admin role in the SLX device.
- Only RS256-based OAuth2 token is supported.
- There is no expiration check in the OAuth2 token.
- You can import only one OAuth2 PKI CA certificate.
- The maximum token length allowed is 1024 bytes.
- The supported token signature algorithm is RSA SHA-256.

SLX Host PKI Certificate Expiry Alerts

An alert is sent when PKI certificates are due to expire.

All configured SLX host PKI certificates are periodically checked for expiration date and time. If a certificate is near expiration, or if already expired, a TRAP is sent, and a RASLOG is logged.

The frequency of alerts is:

- Expiration 60 days or less - weekly alert
- Expiration three days or less - daily alert
- Expired - daily alert

HTTPS Certificates

HTTPS certificate overview

In public key cryptography, each device has a pair of keys: a public key and a private key. These are typically numbers that are chosen to have a specific mathematical relationship.

The private key can be used to create a digital signature for any piece of data using a digital signature algorithm. This typically involves taking a cryptographic hash of the data and operating on it mathematically using the private key. Any device with the public key can check that this signature was created using the private key and the appropriate signature validation algorithm.

SLX-OS supports DSA, RSA and ECDSA encryption keys for HTTPS cryptography. You can generate key pairs, create trust points, and then authenticate and enroll the key pairs into the trust points to obtain the identity certificates.

Configuring HTTPS certificates

In order to support HTTPS, the device needs to be configured with an Identity certificate. This task generates the key pair, then configures the trust points and certificates required for HTTPS security.

When the Apache (web server) boots. it enables HTTPS service only in the presence of HTTPS crypto certificates.

HTTP and HTTPS are mutually exclusive.

The labels for the trust point and the key pair have to be consistent throughout this process.

1. Enter configure terminal mode.

   ```
   device#configure terminal
   ```

2. Generate a key pair (either RSA, ECDSA, or DSA) to sign and encrypt the security payload during the security protocol exchanges with the **crypto key** command.

   ```
   device(config)# crypto key label k1 rsa modulus 2048
   ```

3. Configure a trusted Certificate Authority (CA) so that the imported identity certificate can be verified that it was issued by one of the locally trusted CAs with the **crypto ca** command.

```
device(config)# crypto ca trustpoint t1
device(config-ca-t1)#
```

4. Associate the key pair to the trust point with the **keypair** command. The association between the trust point, key pair, and identity certificate is valid until it is explicitly removed by deleting the certificate, key pair, or trust point.

```
device(config-ca-t1)# keypair k1
```

5. Return to privileged EXEC mode with the **end** command.

```
device(config-ca-t1)# end
```

6. You must authenticate the device to the CA by obtaining the self-signed certificate of the CA with the **crypto ca authenticate** command. Because the certificate of the CA is self-signed, the public key of the CA should be manually authenticated by contacting the CA administrator to compare the fingerprint of the CA certificate.

```
device# crypto ca authenticate t1 cert-type https protocol SCP host 10.70.12.102 user
fvt directory /users/home/
crypto file cacert.pem
Password: **********
```

7. Export the enrollment certificate to the location specified for the remote host with the **crypto ca enroll** command.

```
device# crypto ca enroll t1 cert-type htttps country US state CA locality SJ
organization BRC orgunit SFI common
myhost.extreme.com protocol SCP host 10.70.12.102 user fvt directory /users/home/crypto

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

8. Import the identity certificate from the trust point CA with the **crypto ca import** command. This installs the identity certificate on the device.

```
device# crypto ca import t1 certificate cert-type https protocol SCP host 10.70.12.102
user fvt directory /users/
home/crypto file swcert.pem
Password: **********
```

9. Confirm the configuration with the **show** commands in the example below.

```
device# show crypto ca certificates
Trustpoint: t1
certificate:
SHA1 Fingerprint=B7:5B:DB:9B:24:69:40:39:36:66:4D:59:2C:69:83:8E:93:CA:23:0C
Subject: C=US, ST=CA, L=SJ, O=BRC, OU=SFI, CN=10:00:00:27:F8:87:70:29
Issuer: C=US, ST=CA, L=SJ, O=BR, OU=SF, CN=SOUND/emailAddress=sravi
Not Before: Oct 6 23:44:27 2014 GMT
Not After : Oct 6 23:44:27 2015 GMT
purposes: sslserver
CA certificate:
SHA1 Fingerprint=76:5B:D4:2C:CB:54:FE:6B:C5:E0:E3:FD:11:B0:88:70:80:12:C6:63
Subject: C=US, ST=CA, L=SJ, O=BR, OU=SF, CN=SOUND/emailAddress=sravi
Issuer: C=US, ST=CA, L=SJ, O=BR, OU=SF, CN=SOUND/emailAddress=sravi
Not Before: Sep 19 20:56:49 2014 GMT
Not After : Oct 19 20:56:49 2014 GMT
purposes: sslserver

device# show running-config crypto
crypto key label k1 rsa modulus 2048
crypto ca trustpoint t1
keypair k1
```

10. The HTTP server (either web server or apache server) must be restarted to activate the HTTPS service. Use only one of the following methods:

- If HTTP is in an enabled state (by default HTTP is enabled), then execute the **http server** command to shutdown the service, followed by **no http server** command to enable HTTPS.

- If HTTP is in a disabled state, then execute the **no http server** command to enable HTTPS.

- Reboot the device.

Disabling HTTPS certificates

Disable key pairs and trust points for HTTPS cryptography certificates, which disables the HTTPS security protocol.

To shutdown the HTTPS service without disabling the HTTPS certificates, execute the **http server shutdown** command.

When the Apache (web server) boots. it enables HTTPS service only in the presence of HTTPS crypto certificates.

HTTP and HTTPS are mutually exclusive.

> **Note**
> HTTPS certificates must be configured and enabled for web service to function on the device.

1. Delete the identity device certificate with the **no crypto ca import** command.

```
device# no crypto ca import t1 certificate cert-type https
device# show crypto ca certificates
Trustpoint: t1
certificate: none
CA certificate:
SHA1 Fingerprint=76:5B:D4:2C:CB:54:FE:6B:C5:E0:E3:FD:11:B0:88:70:80:12:C6:63
Subject: C=US, ST=CA, L=SJ, O=BR, OU=SF, CN=SOUND/emailAddress=sravi
Issuer: C=US, ST=CA, L=SJ, O=BR, OU=SF, CN=SOUND/emailAddress=sravi
Not Before: Sep 19 20:56:49 2014 GMT
Not After : Oct 19 20:56:49 2014 GMT
purposes: sslserver
```

2. Unauthenticate the trust point with the **no crypto ca authenticate** command.

```
device# no crypto ca authenticate t1 cert-type https

device# show crypto ca certificates
Certificate Type: none; Trustpoint: t1
Certificate: none
CA certificate(Client authentication): none
CA certificate(Server authentication): none
```

3. Enter configure terminal mode.

```
device#configure terminal
```

4. Disassociate the trust point from the key pair with the **no keypair** command.

```
device(config)# crypto ca trustpoint t1
device(config-ca-t1)#no keypair
device(config-ca-t1)# do show running-config crypto
crypto key label k1 rsa modulus 2048
crypto ca trustpoint t1
  !
!
```

```
device(config-ca-t1)# do show crypto ca trustpoint
Trustpoint: t1; Key-pair: none
```

5. Delete the trust point with the **no crypto ca trustpoint** command.

```
device(config)# no crypto ca trustpoint t1
device(config-ca-t1)# do show running-config crypto
 crypto key label k1 rsa modulus 2048
!
device# show crypto ca trustpoint
Trustpoint: none; Key-pair: none
```

6. Delete the key pair with the **no crypto key** command.

```
device(config-ca-t1)# exit
device(config)#no crypto key label k1
device(config)# do show running-config crypto
% No entries found.

device(config)# do show crypto key mypubkey
key label: none
key type: none
key size: none
```

7. Return to privileged EXEC mode with the **exit** command.

```
device(config-ca-t1)# exit
```

Enabling HTTPS service

After installing the HTTPS certificates, the web server (also known as the apache server) must be restarted to configure the HTTPS service. By default, the web service is running when the device boots.

The HTTPS certificates must be installed.

The web service can be started using one of the following mechanisms:

- Restart the web service by using the **http server use-vrf <vrf-name> shutdown** command in configuration mode, followed by the **no http server use-vrf <vrf-name> shutdown** command.
- Reboot the entire device.
- Commit an HA failover, if that option is available.

Disabling HTTPS service

Disable the HTTPS service using the **http server use-vrf <vrf-name> shutdown** command.

For information on any of the HTTP or HTTPS Server commands, refer to the *Extreme SLX-OS Command Reference*.

Secure Shell

Secure Shell Overview

Secure Shell (SSH) is a protocol that encrypts remote access connections to network devices.

Using encrypted shared keys, SSH authenticates clients or servers, ensuring that the devices accessing your network are authentic.

The steps to configuring SSH are:

- Configure the SSH Server and Client ciphers.
- Configure the SSH Server and Client key-exchange algorithms.
- Configure the SSH Server and Client MACs.
- Configure the maximum number of SSH sessions.

Ciphers, non-CBC ciphers, algorithms, and MACs are not mutually exclusive. Any combination of these items can be configured on the device.

Configure SSH MAC

You can configure SSH Server and Client Message Authentication Codes (MACs).

SSH server must be enabled.

For a complete list of supported MACs, see the online help for the device.

1. Enter global configuration mode.

```
device# configure terminal
```

2. On the SSH server, configure the SSH server MACs.

   You can specify multiple MACs by separating the string names with commas.

   ```
   device(config)# ssh server mac hmac-sha1,hmac-sha2-256,hmac-sha2-512
   ```

3. On the SSH client, configure the SSH client MACs.

   You can specify multiple MACs by separating the string names with commas.

   ```
   device(config)# ssh client mac hmac-sha1,hmac-sha2-256,hmac-sha2-512
   ```

4. Restart the SSH server for the configuration to take effect.

   ```
   device(config)# do ssh-server restart
   Warning: This operation will disconnect all active SSH sessions.

   Are you sure you want to restart the SSH server [y/n]? y
   SSH server is going down for restart NOW !!
   SSH server restarted !!
   ```

5. Confirm the SSH configuration information with one of the following commands.

   ```
   device(config)# do show running-config ssh server
   ssh server mac hmac-sha1,hmac-sha2-256,hmac-sha2-512
   ssh server key rsa 2048
   ssh server key ecdsa 256
   ssh server key dsa

   device(config)# do show running-config ssh client
   ssh client mac hmac-sha1,hmac-sha2-256,hmac-sha2-512

   device(config)# show ssh server status
   SSH Server Rekey Volume: 1024
   SSH Server Auth Tries: 6
   SSH Server Login Timeout: 120
   VRF-Name: mgmt-vrf      Status: Enabled
   VRF-Name: default-vrf   Status: Enabled

   device(config)# do show ssh client status
   SSH Client Mac: hmac-sha1,hmac-sha2-256,hmac-sha2-512
   ```

   > **Note**
   > The `ssh server key dsa` is not supported in FIPS and CC modes.

Configure SSH Ciphers

Use the command line to configure the Secure Shell (SSH) ciphers.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Set the server cipher for SSH.

   You can use multiple ciphers by separating the string names with commas.

   ```
   device(config)# ssh server cipher aes192-cbc,aes128-ctr
   ```

3. Set the client cipher for SSH.

   You can use multiple ciphers by separating the string names with commas.

   ```
   device(config)# ssh client cipher aes192-cbc,aes128-ctr
   ```

4. Restart the SSH server for the configuration to take effect.

   ```
   device(config)# do ssh-server restart
   Warning: This operation will disconnect all active SSH sessions.
   ```

```
Are you sure you want to restart the SSH server [y/n]? y
SSH server is going down for restart NOW !!
SSH server restarted !!
```

5. Confirm the cipher settings with one of the following commands.

```
device(config)# do show running-config ssh server cipher
ssh server cipher aes192-cbc,aes128-ctr

device(config)# do show running-config ssh client cipher
ssh client cipher aes192-cbc,aes128-ctr

device(config)# do show ssh server status
SSH Server Rekey Volume: 1024
SSH Server Auth Tries: 6
SSH Server Login Timeout: 120
VRF-Name: mgmt-vrf       Status: Enabled
VRF-Name: default-vrf    Status: Enabled

device(config)# do show ssh client status
SSH Client Cipher: aes192-cbc,aes128-ctr
```

To see the complete list of supported server or client ciphers, use one of the following commands.

```
device(config)# ssh server cipher ?
Possible completions:
  <string>   Ciphers supported :- aes128-ctr, aes192-ctr, aes256-ctr, aes128-cbc,
aes192-cbc,
             aes256-cbc, aes128-gcm@openssh.com,
             aes256-gcm@openssh.com,3des-cbc,blowfish-cbc,cast128-cbc,arcfour,
arcfour128,
             arcfour256, rijndael-cbc@lysator.liu.se, chacha20-poly1305@openssh.com.
Ciphers
             recommended in FIPS mode :- aes128-ctr, aes192-ctr, aes256-ctr, aes128-
cbc,
             aes192-cbc, aes256-cbc. Configuring anything else is a security risk.
Ciphers
             recommended in CC mode :- aes128-ctr, aes256-ctr, aes128-cbc, aes256-cbc.
             Configuring anything else is a security risk.
device(config)# ssh client cipher ?
Possible completions:
  <string>   Ciphers supported :- aes128-ctr, aes192-ctr, aes256-ctr, aes128-cbc,
aes192-cbc,
             aes256-cbc, aes128-gcm@openssh.com,
             aes256-gcm@openssh.com,3des-cbc,blowfish-cbc,cast128-cbc,arcfour,
arcfour128,
             arcfour256, rijndael-cbc@lysator.liu.se, chacha20-poly1305@openssh.com.
Ciphers
             recommended in FIPS mode :- aes128-ctr, aes192-ctr, aes256-ctr, aes128-
cbc,
             aes192-cbc, aes256-cbc. Configuring anything else is a security risk.
Ciphers
             recommended in CC mode :- aes128-ctr, aes256-ctr, aes128-cbc, aes256-cbc.
             Configuring anything else is a security risk.
```

Remove an SSH Cipher

The "no" form of the **ssh server cipher** and **ssh client cipher** commands sets the SSH ciphers back to the default algorithms.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Use the **ssh server cipher** command to remove the server cipher for SSH.

   You can remove multiple ciphers by separating the string names with commas.

   ```
   device(config)# no ssh server cipher
   ```

3. Use the **ssh client cipher** command to remove the client cipher for SSH.

   You can remove multiple ciphers by separating the string names with commas.

   ```
   device(config)# no ssh client cipher
   ```

Configure SSH Key-exchange

The SSH key-exchange specifies the algorithms used for generating one-time session keys for encryption and authentication with the SSH server.

See the online help on the device for the complete list of supported key exchange algorithms.

For backward compatibility, the string "dh-group-14" is also acceptable in place of "diffie-hellman-group-14-sha1".

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Use the **ssh server key-exchange** command to set the key exchange algorithm for the server.

   You can use multiple key exchange algorithms by separating the string names with commas.

   ```
   device(config)# ssh server key-exchange diffie-hellman-group14-sha1,ecdh-sha2-nistp521
   ```

3. Use the **ssh client key-exchange** command to set the key exchange algorithm for the client.

   You can use multiple key exchange algorithms by separating the string names with commas.

   ```
   device(config)# ssh client key-exchange diffie-hellman-group14-sha1,ecdh-sha2-nistp521
   ```

   The following ssh server and ssh client key exchange algorithms are supported in FIPS mode:
   * ecdh-sha2-nistp256
   * diffie-hellman-group-exchange-sha256
   * diffie-hellman-group14-sha1

   The following ssh server and ssh client key exchange algorithms are supported in CC mode:
   * ecdh-sha2-nistp256
   * diffie-hellman-group14-sha1

4. Restart the SSH server from EXEC mode using the **ssh-server restart** command for the new configuration to take effect.

   ```
   device(config)# exit
   device# ssh-server restart
   ```

Remove an SSH key-exchange Algorithm

The "no" version of the **ssh server key-exchange** command resets the SSH key exchange algorithms back to the default values.

1. Enter configure terminal mode.

   ```
   device#configure terminal
   ```

2. Use the **no ssh server key-exchange** command to reset the key exchange algorithm for the server to the default value.

3. Use the **no ssh client key-exchange** command to reset the key exchange algorithm for the client to the default value.

Removing an SSH MAC

Removes SSH Server and Client Message Authentication Codes (MACs).

The "no" form of the **ssh server mac** and **ssh client mac** commands removes the MACs.

1. Enter configure terminal mode.

```
device# configure terminal
```

2. On the SSH server, enter the **no ssh server mac** command to set the SSH server MACs to default values.

3. On the SSH client, enter the **no ssh client mac** command to set the SSH server MACs to default values.

4. Restart the SSH server from EXEC mode using the **ssh-server restart** command.

```
device# ssh-server restart
```

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 AAAAB3NzaC1yc2EAAAADAQABAAAABAQ
Dnim+Ofjx/id3z2jDxXu9DcMuQqVq/NKi2Lms
+q7dA5Dqww8jlrOGawG8tMySOvnB1ZEvJt1kqNneRi4l6Ot4/7hfd
99rIOPGBP/NJs6xTLUrQhDgxB78ddTg+6euBtkYLTAaTC7kbXGXcO8VVB9+4xrH
+0bkvjU9RRvGJguUfdiFKEfIGVOy
t0atdHi1dmgQ9BE0cO65nc/i9MjMJedBe174/
QT4TxeGeEgaQ57c2AL5It2V4CzrZBDtnixdnHUO5w2vmBR61LZIDVT1
fuX/xYxDAm9H8SDpDX8pZlfFpQBy/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.

Inline SSH Public Key Configuration

SSH password-less authentication supports specification of a public key directly in the command line, instead of importing it from the SSH server.

Previous functionality

Previously, the public key had to be imported from the server by means of an operational CLI, with the following syntax.

```
device# certutil import sshkey user <user> host <host> directory <directory>
file <file> login <login> password <password>
```

The following is a completed command example.

```
device# certutil import sshkey user admin host 10.20.61.151 directory /root/.ssh/
file id_rsa.pub login root password pass
```

SLX-OS first tries to authenticate by using the public key. If it cannot find the public key, it falls back to password-based authentication and allows the user to log in by entering a valid password.

Current functionality

A public key can now be copied directly into the command line, instead of importing it from the server. The syntax is as follows.

```
device# certutil sshkey user <user> pubkey <public key>
```

The following is a complete example.

```
device# certutil sshkey user admin pubkey "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDnim+Ofjx/id3z2jDxXu9DcMuQqVq/NKi2Lms+
q7dA5Dqww8jlrOGawG8tMySOvnB1ZEvJt1kqNneRi4l6Ot4/7hfd99rIOPGBP/NJs6xTLUrQhDgxB78ddTg+
6euBtkYLTAaTC7kbXGXcO8VVB9+4xrH+0bkvjU9RRvGJguUfdiFKEfIGVOyt0atdHi1dmgQ9BE0cO65nc/
i9MjMJedBe174/QT4TxeGeEgaQ57c2AL5It2V4CzrZBDtnixdnHUO5w2vmBR61LZIDVT1fuX/
xYxDAm9H8SDpDX8pZlfFpQBy
/wrkIYPZ/p4OLrUApB/XAJGujrlNlZLEu9U9MPVM/ root@ldap.hc-fusion.in"
```

Note the following conditions:

- The user must exist on the device. By default, there are two users: "admin" and "user". New users can be added by means of the **username** command in global configuration mode.

- The public key must be entered within double quotes (" ").

- Do the following to generate a public key. Run **ssh-keygen -t rsa** on any server from which you want to start an SSH session to the device. Once you run this command, if you have not entered any other path while generating the key, the public key is generated at `/root/.ssh/ id_rsa.pub` by default. Open this file and copy all its contents after the **pubkey** option in the CLI.

- The command to delete the public key remains the same: `device# no certutil sshkey user <user>`

After the public key has been imported or copied by means of the **certutil import sshkey** or the **certutil sshkey** commands, for the specified user, then password-based authentication is disabled for that particular user. The user is not able to log in with a valid password, but password-based authentication continues to work for all other users who do not have the public key configured or imported on the device.

The specified user is allowed to log in only by using a public key. If anyone tries to log in from any other server for which the public key is not present on the device, then the client receives a "Permission denied (publickey)" error message. Once the public key has been removed for the specified user, then password-based authentication is enabled automatically for that particular user.

> **Note**
> Because NETCONF runs over SSH, its behavior is similar to that for SSH.

Note the following conditions:

- Because **certutil** is not a configuration command, the public key configuration is not saved to the configuration file. In order to replay the public key configuration from a file, the following example public keys must be added manually for all the users to the configuration file.

```
do certutil sshkey user test123 pubkey "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDnim+
Ofjx/id3z2jDxXu9DcMuQqVq/NKi2Lms
+q7dA5Dqww8jlrOGawG8tMySOvnB1ZEvJt1kqNneRi4l6Ot4/7hfd99rIOPGBP/
NJs6xTLUrQhDgxB78ddTg+6euBtkYLTAaTC7kbXGXcO8VVB9+4xrH
+0bkvjU9RRvGJguUfdiFKEfIGVOyt0atdHi1dmgQ9BE0cO65nc/
i9MjMJedBe174/QT4TxeGeEgaQ57c2AL5It2V4CzrZBDtnixdnHUO5w2vmBR61LZIDVT1fuX/
xYxDAm9H8SDpDX8pZlfFpQBy/
wrkIYPZ/p4OLrUApB/XAJGujrlNlZLEu9U9MPVM/ root@ldap.hc-fusion.in"
```

You can use **echo** to append this command to the end of the file.

- A restart of the SSH server on all VRF instances occurs whenever the public key is configured or deleted, or a user with a public key is deleted. Therefore, all the existing SSH connections are disconnected.

- Because public authentication is applicable only for SSH, Telnet and REST continue to use the configured password. Access is allowed through Telnet and REST by means of a valid password, even though password-based authentication is disabled for SSH for that user.

NEW! SSH Authentication with x.509 v3 Certificates

SSH provides public key and password authentication methods. With the public key method, the possession of a private key serves as authentication.

The Secure Shell Authentication (SSH) method works by sending a signature created with a private key of the user. The server must check that the key is a valid authenticator for the user and must check that the signature is valid. If both are valid, the authentication request must be accepted. Otherwise, it must be rejected.

In SLX-OS, this feature provides an infrastructure that identifies x.509 v3 certificates and can use the certificates for SSH authentication.

- SSH server and host authentication using x.509 v3 certificates
- SSH user authentication using x.509 v3 certificates
- Dual-mode support for host authentication: public key and x.509 v3 certificate-based support
- Multi-mode support for user authentication: public key, password, and x.509 v3 certificate-based support

The SSH authentication feature supports the x509v3-ssh-rsa and x509v3-rsa20048-sha256 public key algorithms for use in the x.509 v3 certificate-based SSH authentication from x509v3-ssh-dss, x509v3-ssh-rsa,and x509v3-rsa2048-sha256, and the family of algorithms in x509v3-ecdsa-sha2-*. In these algorithms, a public key is stored in an x.509 v3 certificate. This certificate, plus a chain of certificates leading to a trusted certificate authority and optional messages indicating the revocation status of the certificates, is sent as the public key data in the Secure Shell protocol.

The x.509 v3-based public key algorithm (x509v3-ssh-dss, x509v3-ssh-rsa, and x509v3-ecdsa-sha2-*) is performed in the analogous method for the corresponding non-x.509 v3-based public key algorithms (ssh-dss, ssh-rsa, and ecdsa-sha2-*).

The x509v3-rsa2048-sha256 public key algorithm provides a mechanism similar to ssh-rsa, but with a different hash function and additional key size constraints.

Table 33: Related commands

Command	Function
certutil sshx509v3	Configures the SSH user certificate Distinguished Name (DN).
crypto ca authenticate	Identifies the root CA certificate, which is used to sign the Certificate Signing Request (CSR) to generate the server certificate. The **ssh-x509v3** option indicates that the certificate is used for SSH-x509v3 authentication.
crypto ca enroll	Enrolls the trust point by generating the Certificate Signing Request (CSR) and exporting it to the remote certificate server. The **ssh-x509v3** option indicates that the certificate is used for SSH-x509v3 authentication.

Table 33: Related commands (continued)

Command	Function
crypto ca import	Imports the Identity Certificate for security configuration. The **ssh-x509v3** option indicates that the certificate is used for SSH-x509v3 authentication.
crypto import	Imports the Identity Certificate for security configuration. The **ssh-x509v3** option defines the certification protocol.
ssh server algorithm	Configures the SSH server host key algorithm to be used for x.509 v3 certificate-based SSH authentication (server authentication).
ssh server certificate	Configures the SSH server certificate profile and enters SSH server certificate profile configuration mode.
trustpoint sign	Configures the trust point to the server certificate profile that is used to sign the server certificate.

Two Factor SSH Authentication using CAC/PIV Card

Two factor authentication uses a common access card and personal identity verification (CAC/PIV) card in SSH.

Digital certificates in the X.509v3 format (RFC5280) are used to provide identity management for both SSH Client and Server. A chain of signatures by a trusted root certification authority and its intermediate certificate authorities binds a given public signing key to a given digital identity. The term Non-Person-Entity (NPE) is used to describe the certificates assigned to hardware such as web servers, switches, and routers.

For user authentication, the SSH client sends the user's certificate stored on the Personal Identification Verification (PIV) card or Common Access Card (CAC) to the SSH server for verification. The SSH server running on SLX device validates the incoming user certificate using public key infrastructure (PKI) trust-store.

After the validity of the user certificate has been obtained as valid (i.e. not expired or revoked) the second part is to "Authorize" the user and set the privilege level of the user account against LDAP (if configured) or the locally defined user account (if LDAP is not configured). The user name is in the format EPIDI@domain (extracted from subject Common Name) in case of CAC or CHUID@domain (extracted from SAN principal name) in case of PIV. EDIPI is the term that the United States DoD uses and it is a 10-digit number and a subset of CHUID (the term NIST uses). CHUID is a 14 digit number.

If the LDAP/Radius Server is not reachable or it is not configured, the authorization falls back locally on the device for the configured user name.

TLS Server Certificate and Private Key with no Trust Point

NEW! TLS Server Certificate and Private Key with No Trust Point

You can import a TLS server certificate and private key (in PKCS12 format) to an SLX device (with no trust point) and establish a secure connection.

The following is the process for importing the certificate and key and for establishing the secure connection.

* Remove any existing certificate and key from the device.

```
device# no crypto ca import <trustpoint-name> certificate cert-type <https|ssh-x509v3>
device# no crypto ca authenticate <trustpoint-name> cert-type <https|ssh-x509v3>
device# config
Entering configuration mode terminal
device(config)# crypto ca trustpoint <trustpoint-name>
device(config-ca-t1)# no keypair
device(config-ca-t1)# exit
device(config)# no crypto ca trustpoint <trustpoint-name>
device(config)# no crypto key label <key-name>
device(config)# end
device#
```

* Ensure that the certificate to be imported is well formed. In other words, it is a valid signing certificate that has not expired or been tampered with.

  Verify the certificate's creation time with `openssl x509 -noout -text -in tlscert.pem | grep 'Not Before'`. The time on the device must be later than this time or the import will not work.

* Use the **`crypto ca import-pkcs`** command to import the certificate and private key from an external server.

```
crypto ca import-pkcs type <pkcs12> cert-type <ssh-x509v3|https> directory
<dir-name> file <file-name> host <host-name/ip> protocol <SCP|FTP> user <user-name>
password <scp-password> [pkcs-passphrase <pkcs export password>] [use-vrf <vrf name>]
```

* The imported certificate is validated on some parameters to ensure that it is valid. An error message is generated if the certificate is invalid.

  For example, the certificate is validated against the `not before` and `not after time` values. If the time on the device is not within this range, validation fails and an error message is generated.

- Import of an HTTPS certificate is not complete until you do one of the following:
  - Restart the HTTPS server with the following commands:
    ```
    device# configure terminal
    Entering configuration mode terminal
    device(config)# http server use-vrf <VRF Name> sh
    device(config)# no http server use-vrf <VRF Name> sh
    device(config)# end
    device#
    ```
  - Reboot the SLX device.
- Import of an SSHx509v3 certificate is not complete until you perform the following commands:
  ```
  device# configure terminal
  Entering configuration mode terminal
  device(config)# ssh server algorithm hostkey {x509v3-ssh-rsa | x509v3-rsa2048-sha256}
  device(config)# ssh server certificate profile server
  device(ssh-server-cert-profile-server)# trustpoint sign pkcs12
  device(ssh-server-cert-profile-server)# end
  device#
  ```

VXLAN Visibility

VXLAN visibility overview

The Virtual Extensible LAN (VXLAN) visibility feature is used in the transit network devices.

In general, a transit device routes the traffic based on the outer destination virtual tunnel endpoints (VTEP) IP address. However, the VXLAN visibility feature provides a mechanism for deep packet inspection and classifies the packets on the outer Layer 3 header and the VXLAN header and also on the native inner Layer 3 and Layer 4 header.

Extreme's VXLAN visibility has overlay access-control lists (ACLs) of type VXLAN. It is a collection of filters that defines what action to take on the packets thata match the configured parameter in the filter. VXLAN visibility overlay ACLs define filters with parameters that match the outer Layer 3 and Layer 4 VXLAN header and the native inner Layer 3 and Layer 4 fields of a packet.

In addition, a VXLAN overlay ACL accepts a breakout port as a mirror port or a redirect port. You can remove VXLAN visibility rules before changing the breakout port configuration for a port that is being used as a mirror port or a redirect port in the VXLAN rule.

Overlay access list

An overlay access list is a type of access list used in overlay technologies. Currently, Extreme supports overlay access list of type VXLAN transit.

An overlay access list can be associated to the overlay transit.

Type of overlay access lists

There are two types of overlay access lists such as standard overlay access list and extended overlay access list.

The standard overlay access list is available as part of the ternary content-addressable memory (TCAM) profile. Overlay access lists of this type have limited qualifiers and action.

The extended overlay access list is available as part of the ternary content-addressable memory (TCAM) profile. Overlay access lists of this type have extended qualifiers and action.

Limitations and restrictions

The VXLAN visibility has the following limitation and restrictions.

- Only the overlay access control list of type VXLAN transit is supported.
- The rACL and Openflow are not supported as part of vxlan-visibility TCAM profile.
- Rule update of the standard or extended mode is not supported.

Creating an overlay access list

1. From the privileged EXEC mode, enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Use the **overlay access-list type vxlan** command to create an overlay access list.

   Enter the parameter **extended** for creating an overlay access list of type extended or **standard** for creating an overlay access list of type standard.

   ```
   device(config)# overlay access-list type vxlan extended abc_ext
   2016/08/15-23:29:09, [SSMD-1400], 4282, M1 | Active | DCE, INFO, SLX, Overlay access
   list abc_ext is created
   ```

3. Use the **seq** command to insert filtering rules in the overlay access list.

   ```
   device(config-overlay-vxlan-ext-vlx_al)# seq 12 permit dst-vtep-ip-host 10.10.10.1 src-
   vtep-ip-host 20.20.20.1 vni-any sflow count
   2016/08/15-23:29:43, [SSMD-1404], 4283, M1 | Active | DCE, INFO, SLX, Overlay access
   list abc_ext rule sequence number 12 is added.
   ```

Binding overlay access list

You must first create an overlay access list that you want to bind to an overlay transit.

1. Use the **overlay-transit** command to create an overlay transit.

   ```
   device(config)# overlay-transit abc_ext
   ```

2. Use the **overlay access-group** command to bind an overlay access list to an overlay transit.

   ```
   device(config-overlay-transit-vxlan1)# overlay access-group abc_ext in
   2016/08/15-23:30:00, [SSMD-1405], 4284, M1 | Active | DCE, INFO, SLX, Overlay access
   list abc_ext configured on interface Global at Ingress by VXLAN VISIBILITY.
   ```

   Only one overlay access list can be bound to an overlay transit.

3. (Optional) Use the **no overlay overlay access-group** command to unbind an overlay access list.

   ```
   device(config-overlay-transit-vxlan1)# no overlay access-group vlx_al in
   ```

Displaying overlay access list information

Use the following show commands to display the configuration, binding status and statistics pertaining to overlay access list.

1. From the privileged EXEC mode, use the **show access-list overlay transit** command to display which overlay access list is bound with overlay transit.

```
device# show access-list overlay transit tr_name
Overlay Transit Global Binding
  Inbound access-list is abc_ext (From User)
  Outbound access-list is not set
```

2. Use the **show access-list overlay type vxlan** command to display status of individual filters and binding information of the overlay access list.

```
device# show access-list overlay type vxlan acl-name abc_ext
Number of Rules: 4
seq 1000 permit  dst-vtep-ip-host 200.1.1.1 src-vtep-ip-host 150.1.1.1 vni 1 vni-mask
0 redirect Ethernet 2/65 sflow count 44024774(pkts)/52829728800(bytes)
seq 1010 permit  dst-vtep-ip-host 200.1.1.2 src-vtep-ip-host 150.1.1.2 vni 2 vni-mask
0 redirect Ethernet 2/19 sflow count 44024773(pkts)/52829727600(bytes)
seq 1020 permit  dst-vtep-ip-host 200.1.1.3 src-vtep-ip-host 150.1.1.3 vni 3 vni-mask
0 redirect Ethernet 2/43 sflow count 0(pkts)/0(bytes)
seq 1030 permit  dst-vtep-ip-host 200.1.1.4 src-vtep-ip-host 150.1.1.4 vni 4 vni-mask
0 redirect Ethernet 2/67 sflow count 0(pkts)/0(bytes)
Transit : transit_name
```

3. Use the **show statistics access-listcoverlay type vxlan** command to display statistics for specific an overlay access list.

```
device# show statistics access-list overlay type vxlan abc_ext
Number of Rules: 2
seq 1000 permit  dst-vtep-ip-host 200.1.1.1 src-vtep-ip-host 150.1.1.1 vni 1 vni-mask
0 redirect Ethernet 2/65 sflow count 0(pkts)/0(bytes)
seq 1010 permit  dst-vtep-ip-host 200.1.1.2 src-vtep-ip-host 150.1.1.2 vni 2 vni-mask
0 redirect Ethernet 2/19 sflow count 44024773(pkts)/52829727600(bytes)
```

4. Use the **show running-config overlay access-list type vxlan** command to display the overlay access list configuration.

```
device# show running-config overlay access-list type vxlan
overlay access-list type vxlan extended abc_ext
 seq 12 permit dst-vtep-ip-host 12.12.1.1 src-vtep-ip-host 33.4.5.6 vni-any count
sflow native tag none dst-ip-any src-ip-any dst-port-any src-port-any
 seq 123 permit dst-vtep-ip-any src-vtep-ip-any vni-any count!
```

Clearing overlay access list statistics

Use the **clear counters access-list overlay type vxlan** command to remove statistics pertaining to a specific overlay access list.

```
device# clear counters access-list overlay type vxlan abc_ext
```