CONFIGURATION GUIDE

Ruckus FastIron Security Configuration Guide, 08.0.90

Supporting FastIron Software Release 08.0.90

Part Number: 53-1005573-05
Publication Date: 3 September 2019

Copyright, Trademark and Proprietary Rights Information

Export Restrictions

These products and associated technical data (in print or electronic form) may be subject to export control laws of the United States of America. It is your responsibility to determine the applicable regulations and to comply with them. The following notice is applicable for all products or technology subject to export control:

These items are controlled by the U.S. Government and authorized for export only to the country of ultimate destination for use by the ultimate consignee or end-user(s) herein identified. They may not be resold, transferred, or otherwise disposed of, to any other country or to any person other than the authorized ultimate consignee or end-user(s), either in their original form or after being incorporated into other items, without first obtaining approval from the U.S. government or as otherwise authorized by U.S. law and regulations.

Disclaimer

THIS CONTENT AND ASSOCIATED PRODUCTS OR SERVICES ("MATERIALS"), ARE PROVIDED "AS IS" AND WITHOUT WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED. TO THE FULLEST EXTENT PERMISSIBLE PURSUANT TO APPLICABLE LAW, COMMSCOPE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, TITLE, NON-INFRINGEMENT, FREEDOM FROM COMPUTER VIRUS, AND WARRANTIES ARISING FROM COURSE OF DEALING OR COURSE OF PERFORMANCE. CommScope does not represent or warrant that the functions described or contained in the Materials will be uninterrupted or error-free, that defects will be corrected, or are free of viruses or other harmful components. CommScope does not make any warranties or representations regarding the use of the Materials in terms of their completeness, correctness, accuracy, adequacy, usefulness, timeliness, reliability or otherwise. As a condition of your use of the Materials, you warrant to CommScope that you will not make use thereof for any purpose that is unlawful or prohibited by their associated terms of use.

Limitation of Liability

IN NO EVENT SHALL COMMSCOPE, COMMSCOPE AFFILIATES, OR THEIR OFFICERS, DIRECTORS, EMPLOYEES, AGENTS, SUPPLIERS, LICENSORS AND THIRD PARTY PARTNERS, BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, EXEMPLARY OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER, EVEN IF COMMSCOPE HAS BEEN PREVIOUSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, WHETHER IN AN ACTION UNDER CONTRACT, TORT, OR ANY OTHER THEORY ARISING FROM YOUR ACCESS TO, OR USE OF, THE MATERIALS. Because some jurisdictions do not allow limitations on how long an implied warranty lasts, or the exclusion or limitation of liability for consequential or incidental damages, some of the above limitations may not apply to you.

Trademarks

ARRIS, the ARRIS logo, CommScope, Ruckus, Ruckus Wireless, Ruckus Networks, Ruckus logo, the Big Dog design, BeamFlex, ChannelFly, EdgeIron, FastIron, HyperEdge, ICX, IronPoint, OPENG, SmartCell, Unleashed, Xclaim, and ZoneFlex are trademarks of CommScope, Inc. and/or its affiliates. Wi-Fi Alliance, Wi-Fi, the Wi-Fi logo, Wi-Fi Certified, the Wi-Fi CERTIFIED logo, Wi-Fi Protected Access, the Wi-Fi Protected Setup logo, Wi-Fi Protected Setup, Wi-Fi Multimedia and WPA2 and WMM are trademarks or registered trademarks of Wi-Fi Alliance. All other trademarks are the property of their respective owners.

Contents

Preface

Document Conventions

The following table lists the text conventions that are used throughout this guide.

TABLE 1 Text Conventions

Convention	Description	Example
`monospace`	Identifies command syntax examples	`device(config)# interface ethernet 1/1/6`
bold	User interface (UI) components such as screen or page names, keyboard keys, software buttons, and field names	On the **Start** menu, click **All Programs**.
italics	Publication titles	Refer to the *Ruckus Small Cell Release Notes* for more information.

Notes, Cautions, and Warnings

Notes, cautions, and warning statements may be used in this document. They are listed in the order of increasing severity of potential hazards.

NOTE
A NOTE provides a tip, guidance, or advice, emphasizes important information, or provides a reference to related information.

ATTENTION
An ATTENTION statement indicates some information that you must read before continuing with the current action or task.

CAUTION
A CAUTION statement alerts you to situations that can be potentially hazardous to you or cause damage to hardware, firmware, software, or data.

DANGER
A DANGER statement indicates conditions or situations that can be potentially lethal or extremely hazardous to you. Safety labels are also attached directly to products to warn of these conditions or situations.

Command Syntax Conventions

Bold and italic text identify command syntax components. Delimiters and operators define groupings of parameters and their logical relationships.

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

Document Feedback

Ruckus is interested in improving its documentation and welcomes your comments and suggestions.

You can email your comments to Ruckus at #Ruckus-Docs@commscope.com.

When contacting us, include the following information:

- Document title and release number
- Document part number (on the cover page)
- Page number (if appropriate)

For example:

- Ruckus SmartZone Upgrade Guide, Release 5.0
- Part number: 800-71850-001 Rev A
- Page 7

Ruckus Product Documentation Resources

Visit the Ruckus website to locate related documentation for your product and additional Ruckus resources.

Release Notes and other user documentation are available at https://support.ruckuswireless.com/documents. You can locate the documentation by product or perform a text search. Access to Release Notes requires an active support contract and a Ruckus Support Portal user account. Other technical documentation content is available without logging in to the Ruckus Support Portal.

White papers, data sheets, and other product documentation are available at https://www.ruckuswireless.com.

Online Training Resources

To access a variety of online Ruckus training modules, including free introductory courses to wireless networking essentials, site surveys, and Ruckus products, visit the Ruckus Training Portal at https://training.ruckuswireless.com.

Contacting Ruckus Customer Services and Support

The Customer Services and Support (CSS) organization is available to provide assistance to customers with active warranties on their Ruckus products, and customers and partners with active support contracts.

For product support information and details on contacting the Support Team, go directly to the Ruckus Support Portal using https://support.ruckuswireless.com, or go to https://www.ruckuswireless.com and select **Support**.

What Support Do I Need?

Technical issues are usually described in terms of priority (or severity). To determine if you need to call and open a case or access the self-service resources, use the following criteria:

- Priority 1 (P1)—Critical. Network or service is down and business is impacted. No known workaround. Go to the **Open a Case** section.
- Priority 2 (P2)—High. Network or service is impacted, but not down. Business impact may be high. Workaround may be available. Go to the **Open a Case** section.
- Priority 3 (P3)—Medium. Network or service is moderately impacted, but most business remains functional. Go to the **Self-Service Resources** section.
- Priority 4 (P4)—Low. Requests for information, product documentation, or product enhancements. Go to the **Self-Service Resources** section.

Open a Case

When your entire network is down (P1), or severely impacted (P2), call the appropriate telephone number listed below to get help:

- Continental United States: 1-855-782-5871
- Canada: 1-855-782-5871
- Europe, Middle East, Africa, Central and South America, and Asia Pacific, toll-free numbers are available at https://support.ruckuswireless.com/contact-us and Live Chat is also available.
- Worldwide toll number for our support organization. Phone charges will apply: +1-650-265-0903

We suggest that you keep a physical note of the appropriate support number in case you have an entire network outage.

Self-Service Resources

The Ruckus Support Portal at https://support.ruckuswireless.com offers a number of tools to help you to research and resolve problems with your Ruckus products, including:

- Technical Documentation—https://support.ruckuswireless.com/documents

- Community Forums—https://forums.ruckuswireless.com/ruckuswireless/categories
- Knowledge Base Articles—https://support.ruckuswireless.com/answers
- Software Downloads and Release Notes—https://support.ruckuswireless.com/#products_grid
- Security Bulletins—https://support.ruckuswireless.com/security

Using these resources will help you to resolve some issues, and will provide TAC with additional data from your troubleshooting analysis if you still require assistance through a support case or RMA. If you still require help, open and manage your case at https://support.ruckuswireless.com/case_management.

About This Document

What's new in this document

The following table describes changes to this guide for the FastIron 08.0.90b release.

TABLE 2 Summary of changes in FastIron release 08.0.90b

Feature	Description	Described in
Change in SNMP behavior	Accessing RADIUS or TACACS MIB objects via SNMP is enabled by default. Two related commands, **enable snmp configure-radius** and **enable snmp configure-tacacs**, are no longer needed and are deprecated as a result.	The sections related to deprecated behavior and commands have been removed.

The following table includes descriptions of new information added to this guide for the FastIron 08.0.90 release.

TABLE 3 Summary of enhancements in FastIron release 08.0.90

Feature	Description	Described in
MACsec	Support for MACsec is added on ICX 7850-48FS devices.	Media Access Control Security on page 151
TCP MSS	The TCP MSS Adjustment feature is to modify the Maximum Segment Size (MSS) field from the TCP header in the TCP SYN/SYN-ACK packet during the three-way handshake.	TCP MSS Adjustment Overview on page 333
Enable SSH out of the box (and disable telnet)	SSH access to ICX devices is provided out-of-the-box from the factory so that manual intervention is not needed to generate SSH keys, configure a local user, or enable the AAA configurations required for SSH access.	Enable SSH and Disable Telnet Automatically on page 34

Supported hardware

This guide supports the following Ruckus products:

- Ruckus ICX 7850 Series
- Ruckus ICX 7750 Series
- Ruckus ICX 7650 Series
- Ruckus ICX 7450 Series
- Ruckus ICX 7250 Series
- Ruckus ICX 7150 Series

For information about what models and modules these devices support, see the hardware installation guide for the specific product family.

How Command Information is Presented in this Configuration Guide

For all new content supported in FastIron release 08.0.20 and later, command information is documented in a standalone command reference guide.

In the *Ruckus FastIron Command Reference*, the command pages are in alphabetical order and follow a standard format to present syntax, parameters, mode, usage guidelines, examples, and command history.

NOTE
Many commands introduced before FastIron release 08.0.20 are also included in the guide.

Managing User Accounts

Passwords used to secure access

Passwords can be used to secure the following access methods:

- Telnet access can be secured by setting a Telnet password. Refer to Setting a Telnet password on page 19.
- Access to the Privileged EXEC and CONFIG levels of the CLI can be secured by setting passwords for management privilege levels. Refer to Setting passwords for management privilege levels on page 19.

This section also provides procedures for enhancing management privilege levels, recovering from a lost password, and disabling password encryption.

NOTE
You also can configure up to 32 user accounts consisting of a user name and password, and assign each user account a management privilege level. Refer to Local user accounts on page 22.

Setting a Telnet password

By default, the device does not require a user name or password when you log in to the CLI using Telnet.

Assign a password for Telnet access as shown in the following example.

```
device# configure terminal
device(config)# enable telnet password letmein
```

The example assigns the password "letmein" for Telnet access.

Suppressing Telnet connection rejection messages

By default, if a Ruckus device denies Telnet management access to the device, the software sends a message to the denied Telnet client. You have the option to suppress the rejection message. When you enable the option, a denied Telnet client does not receive a message from the Ruckus device. Instead, the denied client simply does not gain access.

To suppress the connection rejection message sent by the device to a denied Telnet client, enter the tel**net server suppress-reject-message** command as shown in the following example.

```
device# configure terminal
device(config)# telnet server suppress-reject-message
```

Setting passwords for management privilege levels

You can set one password for each of the following management privilege levels:

- Super User level - Allows complete read-and-write access to the system. This is generally for system administrators and is the only management privilege level that allows you to configure passwords.
- Port Configuration level - Allows read-and-write access for specific ports but not for global (system-wide) parameters.

- Read-only level - Allows access to the Privileged EXEC mode and User EXEC mode of the CLI but only with read access.

You can assign a password to each management privilege level. You can also configure up to 16 user accounts, consisting of a user name and password, and assign each user account to one of the three privilege levels. Refer to Local user accounts on page 22.

NOTE
You must use the CLI to assign a password for management privilege levels. You cannot assign a password using the Web Management Interface.

If you configure user accounts in addition to privilege level passwords, the device validates a user access attempt using one or both methods (local user account or privilege level password), depending on the order you specify in the authentication-method lists. Refer to Authentication-method lists on page 77.

Perform the following steps to set passwords for management privilege levels.

1. Access global configuration mode.

```
device> enable
device# configure terminal
```

2. Set the Super User level password.

```
device(config)# enable super-user-password test
```

The example sets "test" as the password.

NOTE
You must set the Super User level password before you can set other types of passwords. The Super User level password can be an alphanumeric string, but it cannot begin with a number.

3. Set the Port Configuration level password and the Read-only level passwords.

```
device(config)# enable port-config-password port
device(config)# enable read-only-password read
```

The example sets the Port Configuration level password to "port" and sets "read" as the Read-only level password.

NOTE
If you forget your Super User level password, refer to Recovering from a lost password on page 21.

Augmenting management privilege levels

Each management privilege level provides access to specific areas of the CLI by default:

- Super User level provides access to all commands and displays.
- Port Configuration level gives access to the following levels:
  - The User EXEC
  - Privileged EXEC
  - The port-specific parts of global configuration
  - All interface configuration.
- Read Only level gives access to the following levels:
  - The User EXEC
  - Privileged EXEC.

You can grant additional access to a privilege level on an individual command basis. To grant the additional access, specify the privilege level you are enhancing, the CLI level that contains the command, and the individual command.

NOTE
This feature applies only to management privilege levels of the CLI.

For example, to enhance the Port Configuration privilege level so users also can enter IP commands at the global configuration level, enter the **privilege** command and the parameters shown in the following example.

```
device# configure terminal
device(config)# privilege configure level 4 ip
```

In the example, **configure** specifies that the enhanced access is for a global configuration command. The **level 4** parameter indicates that the enhanced access is for management privilege level 4 (Port Configuration). Executing the command gives the enhanced access to anyone with Port Configuration privileges. The **ip** parameter indicates that the enhanced access is for the IP commands. Users who log in with valid Port Configuration level user names and passwords can enter commands that begin with "ip" at the global configuration level.

For additional information on the **privilege** command and available settings, refer to the *Ruckus FastIron Command Reference*.

Recovering from a lost password

Recovery from a lost password requires direct access to the serial port and a system reset.

NOTE
You can perform this procedure only from the CLI.

Follow the steps given below to recover from a lost password.

1. Start a CLI session over the serial interface to the device.
2. Reboot the device.
3. At the initial boot prompt at system startup, enter **b** to enter the boot monitor mode.
4. Enter **no password** at the prompt. (You cannot abbreviate this command.) This command will cause the device to bypass the system password check.
5. Enter **boot system flash primary** at the prompt.
6. After the console prompt reappears, assign a new password.

Displaying the SNMP community string

If you want to display the SNMP community string, enter the following commands.

```
device# configure terminal
device(config)# enable password-display
device(config)# exit
device# show snmp server
```

The **enable password-display** command enables display of the community string in the output of the **show snmp server** command. Display of the string is still encrypted in the startup-config and running-config files. When the **enable password-display** command is configured, the user password and snmp community string are encrypted in the **show running-config** command output. Enter the command at the global configuration level of the CLI.

Specifying a minimum password length

By default, the Ruckus device imposes no minimum length on the Line (Telnet), Enable, or Local passwords. However, you can configure the device to require that Line, Enable, and Local passwords be at least a specified length. The minimum length can be set to a value from 1 through 48.

For example, to specify that the Line, Enable, and Local passwords contain at least 8 characters, enter the following command.

```
device# configure terminal
device(config)# enable password-min-length 8
```

Local user accounts

You can define up to 32 local user accounts on a Ruckus device. User accounts regulate who can access the management functions in the CLI using the following methods:

- Telnet access
- Web management access
- SNMP access
- SSH access

Local user accounts provide greater flexibility for controlling management access to Ruckus devices than do management privilege level passwords and SNMP community strings of SNMP versions 1 and 2. You can continue to use the privilege level passwords and the SNMP community strings as additional means of access authentication. Alternatively, you can choose not to use local user accounts and instead continue to use only the privilege level passwords and SNMP community strings. Local user accounts are backward-compatible with configuration files that contain privilege level passwords. Refer to Setting passwords for management privilege levels on page 19.

If you configure local user accounts, you also need to configure an authentication-method list for Telnet access, Web management access, and SNMP access. Refer to Authentication-method lists on page 77.

For each local user account, you specify a user name. You also can specify the following parameters:

- A password

  NOTE
  If you use AAA authentication for SNMP access and set the password to be the same as the username, providing the password during authentication is optional. You can provide just the correct username for successful authentication.

- A management privilege level, which can be one of the following:
  - Super User level (default) - Allows complete read-and-write access to the system. This is generally for system administrators and is the only privilege level that allows you to configure passwords.
  - Port Configuration level - Allows read-and-write access for specific ports but not for global parameters.
  - Read Only level - Allows access to the Privileged EXEC mode and User EXEC mode with read access only.
- You can set additional username and password rules. Refer to Enhancements to username and password on page 22.

Enhancements to username and password

This section describes the enhancements to the username and password features introduced in earlier releases.

The following rules are enabled by default:

- Users are required to accept the message of the day.
- Users are locked out (disabled) if they fail to login after three attempts. This feature is automatically enabled. Use the **disable-on-login-failure** command to change the number of login attempts (up to 10) before users are locked out.

The following rules are disabled by default:

- Enhanced user password combination requirements
- User password masking
- Quarterly updates of user passwords
- You can configure the system to store up to 15 previously configured passwords for each user.
- You can use the **disable-on-login-failure** command to change the number of login attempts (up to 10) before users are locked out.
- A password can now be set to expire.

Enabling enhanced user password combination requirements

When strict password enforcement is enabled on the Ruckus device, you must enter a minimum of eight characters containing the following combinations when you create and enable a user password:

- At least two upper case characters
- At least two lower case characters
- At least two numeric characters
- At least two special characters

NOTE
Password minimum and combination requirements are strictly enforced.

Use the **enable strict-password-enforcement** command to enable the password security feature.

```
device# configure terminal
device(config)# enable strict-password-enforcement
```

This feature is disabled by default.

The following security upgrades apply to the **enable strict-password-enforcement** command:

- Passwords must not share four or more concurrent characters with any other password configured on the router. If the user tries to create a password with four or more concurrent characters, the following error message is returned.

```
Error - The substring str within the password has been used earlier, please choose a different password.
```

For example, the previous password was Ma!i4aYa&. The user cannot use any of the following as his or her new password:

- Ma!imai$D because the letters in "Mail" were used consecutively in the previous password
- &3B9aYa& because the letters in "aYa&" were used consecutively in the previous password
- i4aYEv#8 because the letters in "i4aY" were used consecutively in the previous password.

- If the user tries to configure a password that was previously used, the Local User Account configuration is not allowed, and the following message is displayed.

```
This password was used earlier for same or different user, please choose a different password.
```

Enabling user password masking

By default, when you use the CLI to create a user password, the password displays on the console as you type it. For enhanced security, you can configure the Ruckus device to mask the password characters entered at the CLI. When password masking is enabled, the CLI displays asterisks (*) on the console instead of the password characters entered.

The following example shows the default CLI behavior when a username and password is configured.

```
device# configure terminal
device(config)# username kelly password summertime
```

The following example shows the CLI behavior when a username and password are configured with **password-masking** enabled.

```
device# configure terminal
device(config)# username kelly password
Enter Password: ********
```

NOTE
When password masking is enabled, press the **Enter** key before entering the password, and enter the password when prompted.

If **strict-password-enforcement** is enabled, enter a password that contains the required character combination. Refer to Enabling enhanced user password combination requirements on page 23.

To enable password masking, enter the following command.

```
device# configure terminal
device(config)# enable user password-masking
```

Enabling user password aging

For enhanced security, password aging enforces quarterly updates of all user passwords. After 180 days, the CLI automatically prompts users to change their passwords when they attempt to sign on.

When password aging is enabled, the software records the system time that each user password was configured or last changed. The time displays in the output of the **show running-config** command, indicated by set-time.

```
device# show running-config
Current configuration:
....
username waldo password .....
username raveen set-time 2086038248
....
```

The password aging feature uses the NTP server clock to record the set-time. If the network does not have an NTP server, the set-time appears as "set-time 0" in the output of the **show running-config** command.

A username set-time configuration is removed in the following cases:

- The username and password are deleted from the configuration
- The username password expires

When a username set-time configuration is removed, it no longer appears in the **show running-config** output.

NOTE
If a username does not have an assigned password, the username does not have a set-time configuration.

Password aging is disabled by default. To enable it, enter the following commands.

```
device# configure terminal
device(config)# enable user password-aging
```

Configuring password history

By default, the Ruckus device stores the last five user passwords for each user. When changing a user password, the user cannot use any of the five previously configured passwords.

For security purposes, you can configure the Ruckus device to store up to 15 passwords for each user, so that users do not use the same password multiple times. If a user attempts to use a stored password, the system prompts the user to choose a different password.

To configure enhanced password history, enter commands similar to those shown the following example.

```
device# configure terminal
device(config)# enable user password-history 15
```

In the example, the number of previous passwords stored is 15. The *previous-passwords* variable is a value from 1 through 15. The default is 5.

Enhanced login lockout

If a user fails to log in to the device after a configured number of login attempts (by default, 3 attempts), the user is locked out. You can configure the maximum number of invalid login attempts a user can make before being locked out using the **enable user disable-on-login-failure** command. The maximum number of invalid login attempts can be from 1 through 10.

The user account can be configured to automatically re-enable the disabled users using the **enable user** { **disable-on-login-failure** [*invalid-attempts* **login-recovery-time** *recovery-time*] } command. You can specify the recovery time (by default, 3 minutes), after which the locked-out user accounts are re-enabled automatically. The configured recovery time is applicable for all user accounts. The recovery time ranges from 3 through 60 minutes.

If the **login-recovery-time** option is not configured, manual intervention is required to re-enable the locked user account. To manually re-enable a user account, perform one of the following actions:

- Reboot the device to re-enable all locked-out users.
- Execute the **username** *name-string* **enable** command to re-enable a specific user account.

Setting passwords to expire

You can set a user password to expire. Once a password expires, the administrator must assign a new password to the user. To configure a user password to expire, enter commands similar to those shown in the following example.

```
device# configure terminal
device(config)# username sandy expires 20
```

The example sets the password for sandy to expire in 20 days.

The days before expiration can be set to a value from 1 through 365. The default is 90 days.

The expiry details of the user password can be viewed using the **show user** command.

```
device# show user
Username  Password                          Encrypt  Priv  Status   Expire Time
================================================================================
sandy     $1$Gz...uX/$wQ44fVGtsqbKWkQknzAZ6. enabled   0    enabled  20 days
```

Requirement to accept the message of the day

If a message of the day (MOTD) is configured, the user is required to press the Enter key before logging in. The MOTD is configured using the **banner motd** command.

NOTE
Unless configured, the requirement to accept the MOTD is disabled by default.

Local user account configuration

You can create accounts for local users with or without passwords. Accounts with passwords can have encrypted or unencrypted passwords.

You can assign privilege levels to local user accounts, but on a new device, you must create a local user account that has a Super User privilege before you can create accounts with other privilege levels.

NOTE
You must grant the Super User level privilege to at least one account before you add accounts with other privilege levels. You need the Super User account to make further administrative changes.

Local user accounts with no passwords

To create a user account without a password, enter commands similar to those shown in the following example.

```
device# configure terminal
device(config)# username wonka nopassword
```

The example creates the user account wonka without a password.

Local user accounts with unencrypted passwords

If you want to use unencrypted passwords for local user accounts, enter commands similar to those shown in the following example.

```
device# configure terminal
device(config)# username wonka password willy
```

The example sets the password as willy for username wonka

If password masking is enabled, press **Enter** before entering the password as shown in the following example.

```
device# configure terminal
device(config)# username wonka password
Enter Password: *******
```

In the previous example, the password for username wonka is replaced by asterisks because password masking is enabled.

By default, users have full Superuser read-write access. You can control user access by configuring a privilege level.

The **privilege** parameter specifies the privilege level for the account. You can specify one of the following:

- **0** - Super User level (full read-write access)
- **4** - Port Configuration level
- **5** - Read Only level

The default privilege level is **0** . If you want to assign Super User level access to the account, you can enter the command without **privilege 0**.

To create a user account with read-only privileges, enter commands similar to those shown in the following example.

```
device# configure terminal
device(config)# username waldo privilege 5 password whereis
```

The previous example adds a user account for with read-only privileges for waldo and configures the password as whereis. The user waldo can look for information but cannot make configuration changes.

The **password | nopassword** parameter indicates whether the user must enter a password. If you specify **password**, enter the string for the user's password. You can enter up to 48 characters for password-string . If **strict password enforcement** is enabled on the device, you must enter a minimum of eight characters containing the following combinations:

- At least two upper case characters
- At least two lower case characters
- At least two numeric characters
- At least two special characters

NOTE
You must be logged on with Super User access (privilege level 0) to add user accounts or configure other access parameters.

To display user account information, enter the following command.

```
device# show users
```

For more information on the different methods to secure access to the device using the configured username and password, refer to Authentication-method lists on page 77.

Local user accounts with encrypted passwords

You can create encrypted password for local user accounts using the **username** *user-string* **create-password** *password-string* command. By default, the user account encrypted password is encrypted using the MD5 encryption type. You can also configure the password encryption service to encrypt the passwords with different types of encryption, such as SHA1 and SHA256, using the **service password-encryption** command. If the password encryption service type is changed, only the users whose password encryption method matches the newly configured encryption method are allowed to log in. Apart from password encryption, all activities after creating the user account, such as logging in, modifying the local user account, and so on are bound by the configured password encryption service type.

The password encryption methods can be reverted to the default MD5 encryption type by using the **no** form of the **service password-encryption** { **sha1** | **sha256** } command.

Changing a local user password and privilege level

To change a local user password for an existing local user account, enter a command such as the following at the global configuration level of the CLI.

NOTE
You must be logged in with Super User access (privilege level 0) to change user passwords.

```
device(config)# username wonka password willy
```

If password masking is enabled, enter the username, press the Enter key, and then enter the password.

```
device(config)# username wonka password
Enter Password:
```

The above commands change wonka's user name and password.

The password can be up to 48 characters long and must differ from the current password and the two previously configured passwords.

When a password is changed, a message such as the following is sent to the Syslog.

```
SYSLOG: <14>Jan 1 00:00:00 10.44.9.11 Security: Password has been changed for user wonka from console
session.
```

The message includes the name of the user whose password was changed and during which session type, such as console, Telnet, SSH, Web, SNMP, or others.

Using the **username** command, you can use the **create-password** option to create an encrypted password, or you can use the **nopassword** option to modify the user account to log in without a password.

The privilege of the user account can be changed by specifying the privilege levels (**0**, **4**, or **5**).

Preventing unauthorized deletion or modification of a user account

By default, a user account can be deleted or modified without any authentication. Unauthorized deletion or modification of the user account can be prevented using the **service local-user-protection** command. If the user account security is enabled using the **service local-user-protection** command, modification of the password or privilege level of the user is permitted only upon successful validation of the existing user password.

If the **service local-user-protection** command is enabled and you try to modify a user account, you will be prompted for confirmation to proceed. On confirmation, you will be prompted to provide the existing password. The attempt to modify a user account is successful only if correct password is entered.

To prompt the user to confirm existing password before successful password modification, enter the commands such as the following.

```
device(config)# username user1 password xpassx
device(config)# service local-user-protection
device(config)# username user1 password ypasswordy
User already exists. Do you want to modify: (enter 'y' or 'n'): y
To modify or remove user, enter current password: ******
```

Deleting a local user account

You can delete a local user account using the **no username** command. By default, a local user account can be deleted without any authentication.

Unauthorized deletion of the user account can be prevented using the **service local-user-protection** command. If the user account security is enabled using the **service local-user-protection** command, deletion of user accounts is permitted only upon successful validation of the existing user password.

If the **service local-user-protection** command is enabled and you try to delete a user account, you will be prompted for confirmation to proceed. On confirmation, you will be prompted to provide the existing password. The attempt to delete a user account is successful only if correct password is provided.

NOTE
You must be logged in with Super User access (privilege level 0) to delete user accounts.

NOTE
If the last user is removed from the device, the following configuration changes occur:

- AAA configuration specific to local authentication is removed.
- When no other AAA mechanism (RADIUS or TACACS) is configured, "enable aaa console" configuration is removed from the device.
- The method "local" becomes unavailable in the AAA authentication-method list configuration. Refer to Authentication-method lists on page 77 for more information on the authentication-method list.

Remote access to management function restrictions

You can restrict access to management functions from remote sources, including Telnet, the Web Management Interface, and SNMP. The following methods for restricting remote access are supported:

- Using ACLs to restrict Telnet, Web Management Interface, or SNMP access
- Allowing remote access only from specific IP addresses
- Allowing Telnet and SSH access only from specific MAC addresses
- Allowing remote access only to clients connected to a specific VLAN
- Specifically disabling Telnet, Web Management Interface, or SNMP access to the device

The following sections describe how to restrict remote access to a Ruckus ICX device using these methods.

ACL usage to restrict remote access

You can use standard ACLs to control the following access methods to management functions on a Ruckus device:

- Telnet
- SSH
- Web management
- SNMP

Consider the following to configure access control for these management access methods.

1. Configure an ACL with the IP addresses you want to allow to access the device.
2. Configure a Telnet access group, SSH access group, and SNMP community strings. Each of these configuration items accepts an ACL as a parameter. The ACL contains entries that identify the IP addresses that can use the access method.

   The following sections present examples of how to secure management access using ACLs. Refer to the *Rule-Based IP ACLs* chapter for more information on configuring ACLs.

Using an ACL to restrict Telnet access

To configure an ACL that restricts Telnet access to the device, enter commands such as the following.

```
device# configure terminal
device(config)# access-list 10 deny host 10.157.22.32 log
device(config)# access-list 10 deny 10.157.23.0 0.0.0.255 log
device(config)# access-list 10 deny 10.157.24.0 0.0.0.255 log
device(config)# access-list 10 deny 10.157.25.0/24 log
device(config)# access-list 10 permit any
device(config)# telnet access-group 10
device(config)# write memory
```

The previous example configures ACL 10 and applies the ACL as the access list for Telnet access. The device allows Telnet access to all IP addresses except those listed in ACL 10.

NOTE
Standard ACLs can be numbered from 1 through 99.

To configure a more restrictive ACL, create permit entries, and omit the **permit any** entry at the end of the ACL as shown in the following example.

```
device# configure terminal
device(config)# access-list 10 permit host 10.157.22.32
device(config)# access-list 10 permit 10.157.23.0 0.0.0.255
device(config)# access-list 10 permit 10.157.24.0 0.0.0.255
device(config)# access-list 10 permit 10.157.25.0/24
device(config)# telnet access-group 10
device(config)# write memory
```

The ACL in the previous example permits Telnet access only to the IP addresses in the **permit** entries and denies Telnet access from all other IP addresses.

Using an ACL to restrict SSH access

To configure an ACL that restricts SSH access to the device, enter commands such as the following.

```
device(config)#access-list 12 deny host 10.157.22.98 log
device(config)#access-list 12 deny 10.157.23.0 0.0.0.255 log
device(config)#access-list 12 deny 10.157.24.0/24 log
device(config)#access-list 12 permit any
device(config)#ssh access-group 12
device(config)#write memory
```

The num parameter specifies the number of a standard ACL and must be from 1 - 99.

These commands configure ACL 12, then apply the ACL as the access list for SSH access. The device denies SSH access from the IP addresses listed in ACL 12 and permits SSH access from all other IP addresses. Without the last ACL entry for permitting all packets, this ACL would deny SSH access from all IP addresses.

NOTE
In this example, the command **telnet access-group 12** could have been used to apply the ACL configured in the example for Telnet access. You can use the same ACL multiple times.

Using an ACL to restrict Web management access

To configure an ACL that restricts Web management access to the device, enter commands such as the following.

```
device(config)# access-list 12 deny host 209.157.22.98 log
device(config)# access-list 12 deny 209.157.23.0 0.0.0.255 log
device(config)# access-list 12 deny 209.157.24.0/24 log
device(config)# access-list 12 permit any
device(config)# web access-group 12
device(config)# write memory
```

The *num* parameter specifies the number of a standard ACL and must be from 1 – 99. These commands configure ACL 12, then apply the ACL as the access list for Web management access. The device denies Web management access from the IP addresses listed in ACL 12 and permits Web management access from all other IP addresses. Without the last ACL entry for permitting all packets, this ACL would deny Web management access from all IP addresses.

Using ACLs to restrict SNMP access

NOTE
The syntax for using ACLs for SNMP access is different from the syntax for controlling Telnet, SSH, and Web management access using ACLs.

To restrict SNMP access to the device using ACLs, use the **snmp-server community** command in conjunction with **access-list** command entries as shown in the following example.

> **NOTE**
> When **snmp-server community** is configured, all incoming SNMP packets are validated first by their community strings and then by their bound ACLs.

```
device# configure terminal
device(config)# access-list 25 deny host 10.157.22.98 log
device(config)# access-list 25 deny 10.157.23.0 0.0.0.255 log
device(config)# access-list 25 deny 10.157.24.0 0.0.0.255 log
device(config)# access-list 25 permit any
device(config)# access-list 30 deny 10.157.25.0 0.0.0.255 log
device(config)# access-list 30 deny 10.157.26.0/24 log
device(config)# access-list 30 permit any
device(config)# snmp-server community public ro 25
device(config)# snmp-server community private rw 30
device(config)# write memory
```

The example configures ACLs 25 and 30 and applies them to SNMP community strings, which users must enter to gain SNMP access. ACL 25 is used to control read-only (get) access using the "public" community string. ACL 30 is used to control read-write (get/set) access using the "private" community string.

Defining the console idle time

By default, a Ruckus device does not time out serial console sessions. A serial session remains open until you close it. However, you can define how many minutes a serial management session can remain idle before it is timed out.

> **NOTE**
> You must enable AAA support for console commands, AAA authentication, and Exec authorization in order to set the console idle time.

To configure the idle time for a serial console session, use the **console timeout** command as shown in the following example.

```
device# configure terminal
device(config)# console timeout 120
```

The example sets the console timeout to 120 minutes. Possible values are 0 (the default) through 240 minutes.

> **NOTE**
> In RADIUS, the standard attribute Idle-Timeout is used to define the console session timeout value. The Idle-Timeout value is specified in seconds. Within the switch, it is truncated to the nearest minute because the switch configuration is defined in minutes.

Remote access restrictions

By default, a Ruckus device does not control remote management access based on the IP address of the managing device. You can restrict remote management access to a single IP address for the following access methods:

- Telnet access
- SSH access
- Web management access
- SNMP access

In addition, you can restrict all access methods to the same IP address using a single command.

The following examples show the CLI commands for restricting remote access. You can specify only one IP address with each command. However, you can enter each command ten times to specify up to ten IP addresses.

NOTE
You cannot restrict remote management access using the Web Management Interface.

Restricting Telnet access to a specific IP address

You can restrict Telnet access using either an IPv4 or an IPv6 host address. For example, to allow Telnet access to the Ruckus device only to the host with IP address 10.157.22.39, enter the following commands.

```
device# configure terminal
device(config)# telnet client 10.157.22.39
```

Restricting SSH access to a specific IP address

You can restrict SSH access to a specific IPv4 or IPv6 host. For example, to allow SSH access to the Ruckus device only to the host with IP address 10.157.22.39, enter the following command.

```
device# configure terminal
device(config)# ip ssh client 10.157.22.39
```

You can also control SSH access based on the MAC address of the host. For example, to allow SSH access to the Ruckus device for a host with any IP address and MAC address 0000.000f.e910, enter the following command.

```
device# configure terminal
device(config)# ip ssh client any 0000.000f.e910
```

The following example allows access from any host with the IPv6 address 2001::1.

```
device# configure terminal
device(config)# ip ssh client ipv6 2001::1
```

Restricting Web management access to a specific IP address

You can allow Web management access to the Ruckus ICX for a host with a specific IPv4 or IPv6 address. For example, to allow Web management access only to the host with IP address 209.157.22.26, enter the following commands.

```
device# configure terminal
device(config)# web client 209.157.22.26
```

Restricting SNMP access to a specific IP address

You can restrict SNMP access to the Ruckus ICX device to a specific IPv4 or IPv6 host address. For example, to allow SNMP access only to the host with IP address 10.157.22.14, enter the following command.

```
device(config)# snmp-client 10.157.22.14
```

Restricting all remote management access to a specific IP address

You can use the **all-client** command to allow client management access only to a specific IPv4 host address. For example, to allow Telnet, SSH, Web management, and SNMP access to the Ruckus device only to the host with IP address 10.157.22.69, enter the following commands (which is simpler than entering a command for each access type).

```
device# configure terminal
device(config)# all-client 10.157.22.69
```

Restricting access to the device based on IP or MAC address

You can restrict remote management access to the Ruckus device, using Telnet, SSH, HTTP, and HTTPS, based on the connecting client IP or MAC address.

Restricting Telnet connection

You can restrict Telnet connection to a device based on the client IP address or MAC address.

To allow Telnet access to the Ruckus device only to the host with IP address 10.157.22.39 and MAC address 0000.000f.e9a0, enter the following command.

```
device# configure terminal
device(config)# telnet client 10.157.22.39 0000.000f.e9a0
```

The following command allows Telnet access to the Ruckus device to a host with any IP address and MAC address 0000.000f.e9a0.

```
device# configure terminal
device(config)# telnet client any 0000.000f.e9a0
```

Restricting SSH connection

You can restrict SSH connection to a device based on the client IP address or MAC address.

To allow SSH access to the Ruckus device only to the host with IP address 10.157.22.39 and MAC address 0000.000f.e9a0, enter the following commands.

```
device# configure terminal
device(config)#ip ssh client 10.157.22.39 0000.000f.e9a0
```

To allow SSH access to the Ruckus device to a host with any IP address and MAC address 0000.000f.e9a0, enter the following commands.

```
device# configure terminal
device(config)# ip ssh client any 0000.000f.e9a0
```

Defining the Telnet idle time

You can define how many minutes a Telnet session can remain idle before it is timed out. An idle Telnet session is a session that is still sending TCP ACKs in response to keepalive messages from the device but is not being used to send data.

To configure the idle time for a Telnet session, enter the **telnet timeout** command followed by the number of minutes of idle time as shown in the following example.

```
device# configure terminal
device(config)# telnet timeout 120
```

The example configures the idle time for a Telnet session as 120 minutes. Valid values are 0 through 240. The default value is 0 (no timeout; the session remains up).

Changing the login timeout period for Telnet sessions

By default, the login timeout period for a Telnet session is 1 minute. To change the login timeout period, enter the **telnet login-timeout** command followed by the number of minutes as shown in the following example.

```
device# configure terminal
device(config)# telnet login-timeout 5
```

The example sets the login timeout to five minutes. The timeout can be set to a value of 1 through 10 minutes.

Specifying the maximum number of login attempts for Telnet access

> **NOTE**
> You must configure Telnet with the **enable telnet authentication local** command to enable only a certain number of Telnet login attempts.

If you are connecting to the Ruckus device using Telnet, the device prompts you for a username and password. By default, you have up to 4 chances to enter a correct username and password. If you do not enter a correct username or password after 4 attempts, the Ruckus device disconnects the Telnet session.

To specify the number of attempts a Telnet user has to enter a correct username and password, enter the **telnet login-retries** command followed by a value from 0 through 5 as shown in the following example.

```
device# configure terminal
device(config)# telnet login-retries 5
```

The example sets the number of Telnet login attempts allowed to 5.

Enable SSH and Disable Telnet Automatically

The Enable SSH and Disable Telnet Automatically feature provides SSH access to the device out-of-the-box without the manual intervention necessary to generate SSH keys, configure a local user, and enable AAA configurations that are required for SSH access.

The Enable SSH and Disable Telnet Automatically feature is applicable if the device does not have the startup configuration file in the flash memory during an initial bootup. This feature is disabled if the device obtains the configurations automatically through DHCP auto provisioning or from SmartZone. SmartZone can push the configuration file to the device. Enable SSH and Disable Telnet Automatically is not applicable in FIPS or FIPS-CC mode. The syslog messages generated by SmartZone are suppressed.

Enable SSH and Disable Telnet Automatically creates an RSA-2048 key in the device during bootup if an SSH key does not exist.

If the device does not have a startup configuration, console authentication is enabled. Upon initial access to the device from any session (CLI, SSH, or web), the access is allowed with a default local username and default password. After user authentication has succeeded, the password for the default local user must be modified. The Ruckus ICX device automatically obtains an IP address through DHCP, have a record of this IP address before accessing the device from SSH or web sessions.

> **NOTE**
> SmartZone connectivity and ICX-M functionality are not impacted by the automatic disabling of Telnet.

Console Access after device boots up without any startup configuration file

```
User Access Verification
Please Enter Login Name:super
Please Enter Password:sp-admin
```

```
User login successful.
User 'super' login successful with default password. Please change the password.
Enter new password for user 'super':
Reconfirm new password for user 'super':
Password modified successfully for user 'super'.
Authentication is enabled in the device for Console/WEB/SSH
```

Default Username and Password

When SSH is enabled out of the box, use the following default username and password to initially access the ICX device:

- default local username: super
- default password: sp-admin

Bootup Configuration Details for Enable SSH and Disable Telnet Automatically

The following configurations are added to the device during bootup if FIPS is not enabled and the devices does not have a startup configuration file in the flash memory:

```
aaa authentication web-server default local
aaa authentication login default local
enable aaa console
no telnet server
username super password sp-admin
```

- The added configurations are removed automatically from the device if the running configuration file is downloaded by way of DHCP auto-provisioning before being accessed by way of the CLI, SSH, or the web.
- The added configurations are removed automatically from the device if the device connects to SmartZone (SZ) before being accessed by way of the CLI, SSH, or the web.
- The device allows initial access only after using the default local username (super) and password (sp-admin).
- You must change the password for the default local user (super) after accessing the device with the default credentials. The new password is not autosaved to the startup configuration. You must use the **write memory** command after the first login to save the configuration across reloads.
- Console authentication is enabled on the device, so you will be prompted for the username and password each time you attempt to log in.
- After you obtain access to the device, you can change any configurations on the device (such as removing the super user or removing automatically added AAA configurations) and use the **write memory** command to save the modified configurations across reloads.

  NOTE
  Access to the device is not possible if the username is deleted after initial access and the write memory command is used without removing the AAA configurations on the device. The only way to recover the device is to perform a factory reset or remove the startup configuration from the boot time.

Restricting remote access to the device to specific VLAN IDs

You can restrict management access to a Ruckus device to ports within a specific port-based VLAN. VLAN-based access control applies to the following access methods:

- Telnet access
- Web management access

- SNMP access
- TFTP access

By default, access is allowed for all these methods on all ports. Once you configure security for a given access method based on VLAN ID, access to the device using that method is restricted to only the ports within the specified VLAN.

VLAN-based access control works in conjunction with other access control methods. For example, suppose you configure an ACL to permit Telnet access only to specific client IP addresses, and you also configure VLAN-based access control for Telnet access. In this case, the only Telnet clients that can access the device are clients that have one of the IP addresses permitted by the ACL and are connected to a port that is in a permitted VLAN. Clients that have a permitted IP address but are connected to a port in a VLAN that is not permitted still cannot access the device through Telnet.

Restricting Telnet access to a specific VLAN

To allow Telnet access only to clients in a specific VLAN, enter the **telnet server enable vlan** command followed by the VLAN ID as shown in the following example.

```
device# configure terminal
device(config)# telnet server enable vlan 10
```

The example configures the device to allow Telnet management access only to clients connected to ports within port-based VLAN 10. Clients connected to ports that are not in VLAN 10 are denied management access.

Restricting Web management access to a specific VLAN

To allow Web management access only to clients in a specific VLAN, enter the **web-management enable vlan** command followed by the VLAN ID as shown in the following example.

```
device# configure terminal
device(config)# web-management enable vlan 10
```

The example configures the device to allow Web management access only to clients connected to ports within port-based VLAN 10. Clients connected to ports that are not in VLAN 10 are denied management access.

NOTE
You can also configure Web management access for an Ethernet port or a range of ports. Refer to the *Ruckus FastIron Command Reference* for more information on the **web-management** command.

Restricting SNMP access to a specific VLAN

To allow SNMP access only to clients in a specific VLAN, enter the **snmp-server enable vlan** command followed by the VLAN ID as shown in the following example.

```
device# configure terminal
device(config)# snmp-server enable vlan 40
```

The command in this example configures the device to allow SNMP access only to clients connected to ports within port-based VLAN 40. Clients connected to ports that are not in VLAN 40 are denied access.

NOTE
You can also configure SNMP access for a specific Ethernet port or range of ports. Refer to the *Ruckus FastIron Command Reference* for more information on the **snmp-server enable** command.

Restricting TFTP access to a specific VLAN

To allow TFTP access only to clients in a specific VLAN, enter the **tftp client enable vlan** command followed by the VLAN ID as shown in the following example.

```
device# configure terminal
device(config)# tftp client enable vlan 40
```

The example configures the device to allow TFTP access only to clients connected to ports within port-based VLAN 40. Clients connected to ports that are not in VLAN 40 are denied access.

Designated VLAN for management sessions to a Layer 2 switch

All Ruckus ICX devices support the creation of management VLANs. By default, the management IP address you configure on a Layer 2 Switch applies globally to all the ports on the device. This is true even if you divide the device ports into multiple port-based VLANs.

If you want to restrict the IP management address to a specific port-based VLAN, you can make that VLAN the designated management VLAN for the device. When you configure a VLAN to be the designated management VLAN, the management IP address you configure on the device is associated only with the ports in the designated VLAN. To establish a management session with the device, a user must access the device through one of the ports in the designated VLAN.

You can also configure up to five default gateways for the designated VLAN and associate a metric with each one. The software uses the gateway with the lowest metric. The other gateways reside in the configuration but are not used. To use one of the other gateways, modify the configuration so that the gateway you want to use has the lowest metric.

If more than one gateway has the lowest metric, the gateway that appears first in the running-config is used.

NOTE
On ICX 7750, ICX 7450, and ICX 7250 devices, pings to the data port in a VLAN are not supported if the management VLAN is not configured on the VLAN.

NOTE
If you have already configured a default gateway globally and you do not configure a gateway in the VLAN, the software uses the globally configured gateway and gives the gateway a metric value of 1.

To configure a designated management VLAN, enter commands such as those shown in the following example.

```
device# configure terminal
device(config)# vlan 10 by port
device(config-vlan-10)# untag ethernet 1/1/1 to 1/1/4
device(config-vlan-10)# management-vlan
device(config-vlan-10)# default-gateway 10.10.10.1 1
device(config-vlan-10)# default-gateway 10.20.20.1 2
```

The example configures port-based VLAN 10 to consist of ports 1/1/1 through 1/1/4 and to be the designated management VLAN. Two default gateways are configured for the VLAN. Since the 10.10.10.1 gateway is configured with a metric of 1, the software uses this gateway. The other gateway remains in the configuration but is not used. To use the 10.20.20.1 gateway, which has a metric of 2, reconfigure it so that it has a lower metric than the other gateway. The metric for a gateway can be set to a value of 1 through 5. There is no default.

Device management security

By default, all management access is disabled. Each of the following management access methods must be specifically enabled as required in your installation:

- SSHv2
- SNMP
- Web management through HTTP
- Web management through HTTPS

The commands for granting access to each of these management interfaces is described in the following sections.

Allowing SSHv2 access to the ICX device

To allow SSHv2 access to the Ruckus device, you must generate a Crypto Key as shown in the following example.

```
device# configure terminal
device(config)# crypto key generate
```

The example generates a DSA key pair.

In addition, you must use AAA authentication to create a password to allow SSHv2 access as shown in the following example.

```
device# configure terminal
device(config)# aaa authentication login default tacacs+ local
```

The example configures AAA authentication to use TACACS+ for authentication as the default or local authentication if TACACS+ is not available.

Allowing SNMP access to the ICX device

To allow SNMP access to the Ruckus device, enter the **snmp-server** command as shown in the following example.

```
device# configure terminal
device(config)# snmp-server
```

Allowing Web management through HTTP for the ICX device

To allow web management through HTTP for the Ruckus ICX device, enable web management as shown in the following example.

```
device# configure terminal
device(config)# web-management http
```

Use the **https** parameter to enable web management through HTTPS. For information on generating requisite certificates for HTTPS access, refer to Allowing Web management through HTTPS on page 38.

Using the **web-management** command without the http or https option makes web management available for both.

Allowing Web management through HTTPS

To allow web management through HTTPS, you must enable web management as shown in Allowing Web management through HTTP for the ICX device on page 38. You must also generate a crypto SSL certificate or import digital certificates issued by a third-party Certificate Authority (CA).

Generate a crypto SSL certificate as shown in the following example.

```
device# configure terminal
device(config)# crypto-ssl certificate generate
```

You can import a digital certificate issued by a third-party Certificate Authority (CA) and save it in the flash memory, as shown in the following example. Enter the **ip ssl certificate-data-file tftp** command followed by the IP address of the TFTP server from which the digital certificate is to be downloaded and the filename of the certificate to be imported.

```
device# configure terminal
device(config)# ip ssl certificate-data-file tftp 10.10.10.1 cacert.pem
```

The example downloads the digital certificate file cacert.pem from the TFTP server with the IP address 10.10.10.1.

Disabling specific access methods

You can specifically disable the following access methods:

- Telnet access
- Web management access
- SNMP access
- TFTP

NOTE
If you disable Telnet access, you will not be able to access the CLI except through a serial connection to the management module. If you disable SNMP access, you will not be able to use an SNMP-based management application.

Disabling Telnet access

You can use a Telnet client to access the CLI of the device over the network. If you do not plan to use the CLI over the network and want to disable Telnet access to prevent others from establishing CLI sessions with the device, enter the following command.

```
device# configure terminal
device(config)# no telnet server
```

To re-enable Telnet operation, enter the following command.

```
device# configure terminal
device(config)# telnet server
```

Disabling Web management access

If you want to prevent access to the device through the Web Management Interface, you can disable the Web Management Interface.

NOTE
As soon as you make this change, the device stops responding to Web management sessions. If you make this change using your Web browser, your browser can contact the device, but the device will not reply once the change takes place.

To disable the Web Management Interface, enter the following command.

```
device# configure terminal
device(config)# no web-management
```

Use the **no web-management** command with no option specified to disable both web management through http access and web management through https access.

Use the command **no web-management http** to disable only web management through http access.

Use the command **no web-management https** to disable only web management through https access.

Disabling Web management access by HP ProCurve Manager

By default, TCP ports 80 and 280 are enabled on the Ruckus ICX device. TCP port 80 (HTTP) allows access to the device Web Management Interface. TCP port 280 allows access to the device by HP ProCurve Manager.

The **no web-management** command disables both TCP ports. However, if you want to disable only port 280 and leave port 80 enabled, use the **hp-top-tools** option as shown in the following example.

```
device# configure terminal
device(config)# no web-management hp-top-tools
```

The example disables TCP port 280.

Disabling SNMP access

To disable SNMP management of the device, enter the **no snmp-server** command as shown in the following example.

```
device# configure terminal
device(config)# no snmp-server
```

To re-enable SNMP management of the device, enter the **snmp-server** command.

Disabling TFTP access

By default, TFTP client access is enabled. You can globally disable TFTP to block TFTP client access.

To disable TFTP client access, enter the **tftp disable** command as shown in the following example.

```
device# configure terminal
device(config)# tftp disable
```

When TFTP is disabled, users are prohibited from using the **copy tftp** command to copy files to the system flash. If you enter the **copy tftp** command while TFTP is disabled, the system will reject the command and display an error message.

Re-enable TFTP client access after it has been disabled as shown in the following example.

```
device# configure terminal
device(config)# no tftp disable
```

TACACS+ Server Authentication

TACACS and TACACS+ security

You can use the security protocol Terminal Access Controller Access Control System (TACACS) or TACACS+ to authenticate the following kinds of access to the Ruckus device:

- Telnet access
- SSH access
- Console access
- Web management access
- Access to the Privileged EXEC level and CONFIG levels of the CLI

The TACACS and TACACS+ protocols define how authentication, authorization, and accounting information is sent between a Ruckus device and an authentication database on a TACACS/TACACS+ server. TACACS/TACACS+ services are maintained in a database, typically on a UNIX workstation or PC with a TACACS/TACACS+ server running.

How TACACS+ differs from TACACS

TACACS is a simple UDP-based access control protocol originally developed by BBN for MILNET. TACACS+ is an enhancement to TACACS and uses TCP to ensure reliable delivery.

TACACS+ is an enhancement to the TACACS security protocol. TACACS+ improves on TACACS by separating the functions of authentication, authorization, and accounting (AAA) and by encrypting all traffic between the Ruckus device and the TACACS+ server. TACACS+ allows for arbitrary length and content authentication exchanges, which allow any authentication mechanism to be utilized with the Ruckus device. TACACS+ is extensible to provide for site customization and future development features. The protocol allows the Ruckus device to request very precise access control and allows the TACACS+ server to respond to each component of that request.

NOTE
TACACS+ provides for authentication, authorization, and accounting, but an implementation or configuration is not required to employ all three.

TACACS/TACACS+ authentication, authorization,and accounting

When you configure a Ruckus device to use a TACACS/TACACS+ server for authentication , the device prompts users who are trying to access the CLI for a user name and password, then verifies the password with the TACACS/TACACS+ server.

If you are using TACACS+, Ruckus recommends that you also configure authorization , in which the Ruckus device consults a TACACS+ server to determine which management privilege level (and which associated set of commands) an authenticated user is allowed to use. You can also optionally configure accounting , which causes the Ruckus device to log information on the TACACS+ server when specified events occur on the device.

NOTE
By default, a user logging into the device from Telnet or SSH would first enter the User EXEC level. The user can enter the **enable** command to get to the Privileged EXEC level. A user that is successfully authenticated can be automatically placed at the Privileged EXEC level after login. Refer to Entering privileged EXEC mode after a Telnet or SSH login on page 49.

Configuring TACACS/TACACS+ for devices in a Ruckus traditional stack

Because devices operating in a Ruckus traditional stack topology present multiple console ports, you must take additional steps to secure these ports when configuring TACACS/TACACS+.

The following is a sample AAA console configuration using TACACS+.

```
aaa authentication login default tacacs+ enable
aaa authentication login privilege-mode
aaa accounting commands 0 default start-stop  tacacs+
aaa accounting exec default start-stop  tacacs+
aaa accounting system default start-stop  tacacs+
ip address 10.10.6.56/255
tacacs-server host 255.253.255
tacacs-server key 2 $d3NpZ0BVXFpJ
```

TACACS authentication

NOTE
Also, multiple challenges are supported for TACACS+ login authentication.

When TACACS authentication takes place, the following events occur.

1. A user attempts to gain access to the Ruckus device by doing one of the following:
   - Logging into the device using Telnet, SSH, or the Web Management Interface
   - Entering the Privileged EXEC level or CONFIG level of the CLI
2. The user is prompted for a username and password.
3. The user enters a username and password.
4. The Ruckus device sends a request containing the username and password to the TACACS server.
5. The username and password are validated in the TACACS server database.
6. If the password is valid, the user is authenticated.

TACACS+ authentication

When TACACS+ authentication takes place, the following events occur.

1. A user attempts to gain access to the Ruckus device by doing one of the following:
   - Logging into the device using Telnet, SSH, or the Web Management Interface
   - Entering the Privileged EXEC level or CONFIG level of the CLI
2. The user is prompted for a username.
3. The user enters a username.
4. The Ruckus device obtains a password prompt from a TACACS+ server.
5. The user is prompted for a password.
6. The user enters a password.

7. The Ruckus device sends the password to the TACACS+ server.
8. The password is validated in the TACACS+ server database.
9. If the password is valid, the user is authenticated.

TACACS+ authorization

Ruckus devices support two kinds of TACACS+ authorization:

- Exec authorization determines a user privilege level when they are authenticated
- Command authorization consults a TACACS+ server to get authorization for commands entered by the user

When TACACS+ exec authorization takes place, the following events occur.

1. A user logs into the Ruckus device using Telnet, SSH, or the Web Management Interface
2. The user is authenticated.
3. The Ruckus device consults the TACACS+ server to determine the privilege level of the user.
4. The TACACS+ server sends back a response containing an A-V (Attribute-Value) pair with the privilege level of the user.
5. The user is granted the specified privilege level.

When TACACS+ command authorization takes place, the following events occur.

1. A Telnet, SSH, or Web Management Interface user previously authenticated by a TACACS+server enters a command on the Ruckus device.
2. A Telnet, SSH, or Web Management Interface user previously authenticated by a TACACS+server enters a command on the Ruckus device.
3. The Ruckus device looks at its configuration to see if the command is at a privilege level that requires TACACS+ command authorization.
4. If the command belongs to a privilege level that requires authorization, the Ruckus device consults the TACACS+ server to see if the user is authorized to use the command.
5. If the user is authorized to use the command, the command is executed.

TACACS+ accounting

TACACS+ accounting works as follows.

1. One of the following events occur on the Ruckus device:
   - A user logs into the management interface using Telnet or SSH
   - A user enters a command for which accounting has been configured
   - A system event occurs, such as a reboot or reloading of the configuration file
2. The Ruckus device checks the configuration to see if the event is one for which TACACS+ accounting is required.
3. If the event requires TACACS+ accounting, the Ruckus device sends a TACACS+ Accounting Start packet to the TACACS+ accounting server, containing information about the event.
4. The TACACS+ accounting server acknowledges the Accounting Start packet.
5. The TACACS+ accounting server records information about the event.
6. When the event is concluded, the Ruckus device sends an Accounting Stop packet to the TACACS+ accounting server.
7. The TACACS+ accounting server acknowledges the Accounting Stop packet.

AAA operations for TACACS/TACACS+

The following table lists the sequence of authentication, authorization, and accounting operations that take place when a user gains access to a Ruckus device that has TACACS/TACACS+ security configured.

User action	Applicable AAA operations
User attempts to gain access to the Privileged EXEC and CONFIG levels of the CLI	Enable authentication: aaa authentication enable default method-list
	Exec authorization (TACACS+): aaa authorization exec default tacacs+
	System accounting start (TACACS+): aaa accounting system default start-stop method-list
User logs in using Telnet/SSH	Login authentication: aaa authentication login default method-list
	Exec authorization (TACACS+): aaa authorization exec default tacacs+
	Exec accounting start (TACACS+): aaa accounting exec default method-list
	System accounting start (TACACS+): aaa accounting system default start-stop method-list
User logs into the Web Management Interface	Web authentication: aaa authentication web-server default <method-list>
	Exec authorization (TACACS+): aaa authorization exec default tacacs+
User logs out of Telnet/SSH session	Command accounting (TACACS+): aaa accounting commands privilege-level default start-stop method-list
	EXEC accounting stop (TACACS+): aaa accounting exec default start-stop method-list
User enters system commands (for example, reload , boot system)	Command authorization (TACACS+): aaa authorization commands privilege-level default method-list
	Command accounting (TACACS+): aaa accounting commands privilege-level default start-stop method-list
	System accounting stop (TACACS+): aaa accounting system default start-stop method-list
User enters the command: [no] aaa accounting system defaultstart-stop method-list	Command authorization (TACACS+): aaa authorization commands privilege-level default method-list
	Command accounting (TACACS+): aaa accounting commands privilege-level default start-stop method-list
	System accounting start (TACACS+): aaa accounting system default start-stop method-list

AAA security for commands pasted into the running-config

If AAA security is enabled on the device, commands pasted into the running-config are subject to the same AAA operations as if they were entered manually.

When you paste commands into the running-config, and AAA command authorization or accounting, or both, are configured on the device, AAA operations are performed on the pasted commands. The AAA operations are performed before the commands are actually added to the running-config. The server performing the AAA operations should be reachable when you paste the commands into the running-config file. If the device determines that a pasted command is invalid, AAA operations are halted on the remaining commands. The remaining commands may not be executed if command authorization is configured.

TACACS/TACACS+ configuration considerations

- You must deploy at least one TACACS/TACACS+ server in your network.

- Ruckus devices support authentication using up to eight TACACS/TACACS+ servers. The device tries to use the servers in the order you add them to the device configuration.
- You can select only one primary authentication method for each type of access to a device (CLI through Telnet, CLI Privileged EXEC and CONFIG levels). For example, you can select TACACS+ as the primary authentication method for Telnet CLI access, but you cannot also select RADIUS authentication as a primary method for the same type of access. However, you can configure backup authentication methods for each access type.
- You can configure the Ruckus device to authenticate using a TACACS or TACACS+ server, but not both.

Identifying the TACACS/TACACS+ servers

To use TACACS/TACACS+ servers to authenticate access to a Ruckus device, you must identify the servers to the Ruckus device.

For example, to identify three TACACS/TACACS+ servers, enter commands such as the following.

```
device(config)#tacacs-server host 10.94.6.161
device(config)#tacacs-server host 10.94.6.191
device(config)#tacacs-server host 10.94.6.122
```

The ip-addr | ipv6-addr | hostname parameter specifies the IP address or host name of the server. You can enter up to eight **tacacs-server host** commands to specify up to eight different servers.

> **NOTE**
> To specify the server's host name instead of its IP address, you must first identify a DNS server using the **ip dns server-address** ip-addr command at the global CONFIG level.

If you add multiple TACACS/TACACS+ authentication servers to the Ruckus device, the device tries to reach them in the order you add them. For example, if you add three servers in the following order, the software tries the servers in the same order.

- 10.94.6.161
- 10.94.6.191
- 10.94.6.122

You can remove a TACACS/TACACS+ server by entering **no** followed by the **tacacs-server** command. For example, to remove 10.94.6.161, enter the following command.

```
device(config)#no tacacs-server host 10.94.6.161
```

> **NOTE**
> If you erase a **tacacs-server** command (by entering "no" followed by the command), make sure you also erase the **aaa** commands that specify TACACS/TACACS+ as an authentication method. (Refer to Configuring authentication-method lists forTACACS and TACACS+ on page 47.) Otherwise, when you exit from the CONFIG mode or from a Telnet session, the system continues to believe it is TACACS/TACACS+ enabled and you will not be able to access the system.

The **auth-port** parameter specifies the UDP (for TACACS) or TCP (for TACACS+) port number of the authentication port on the server. The default port number is 49.

Specifying different servers for individual AAA functions

Separate TACACS+ servers can be configured and assigned for specific AAA tasks. For example, you can designate one TACACS+ server to handle authorization and another TACACS+ server to handle accounting. You can set the TACACS+ key for each server.

To specify different TACACS+ servers for authentication, authorization, and accounting, enter the command such as following.

```
device(config)#tacacs-server host 10.2.3.4 auth-port 49 authentication-only key abc
device(config)#tacacs-server host 10.2.3.5 auth-port 49 authorization-only key def
device(config)#tacacs-server host 10.2.3.6 auth-port 49 accounting-only key ghi
```

The default parameter causes the server to be used for all AAA functions.

After authentication takes place, the server that performed the authentication is used for authorization and accounting. If the authenticating server cannot perform the requested function, then the next server in the configured list of servers is tried; this process repeats until a server that can perform the requested function is found, or every server in the configured list has been tried.

Setting optional TACACS and TACACS+ parameters

You can set the following optional parameters in a TACACS and TACACS+ configuration:

- TACACS+ key - This parameter specifies the value that the Ruckus device sends to the TACACS+ server when trying to authenticate user access.
- Retransmit interval - This parameter specifies how many times the Ruckus device will resend an authentication request when the TACACS/TACACS+ server does not respond. The retransmit value can be from 1 - 5 times. The default is 3 times.
- Dead time - This parameter specifies how long the Ruckus device waits for the primary authentication server to reply before deciding the server is dead and trying to authenticate using the next server. The dead-time value can be from 1 - 5 seconds. The default is 3 seconds.
- Timeout - This parameter specifies how many seconds the Ruckus device waits for a response from a TACACS/TACACS+ server before either retrying the authentication request, or determining that the TACACS/TACACS+ servers are unavailable and moving on to the next authentication method in the authentication-method list. The timeout can be from 1 - 15 seconds. The default is 3 seconds.

Setting the TACACS+ key

The **key** parameter in the **tacacs-server** command is used to encrypt TACACS+ packets before they are sent over the network. The value for the **key** parameter on the Ruckus device should match the one configured on the TACACS+ server. The key can be from 1 - 32 characters in length and cannot include any space characters.

> **NOTE**
> The **tacacs-server key** command applies only to TACACS+ servers, not to TACACS servers. If you are configuring TACACS, do not configure a key on the TACACS server and do not enter a key on the Ruckus device.

To specify a TACACS+ server key, enter a command such as following.

```
device(config)#tacacs-server key rkwong
```

When you display the configuration of the Ruckus device, the TACACS+ keys are encrypted. For example.

```
device(config)#
tacacs-server key abc
device(config)#write terminal
...
tacacs-server host 10.2.3.5 auth-port 49
tacacs key 2$!2d
```

NOTE
Encryption of the TACACS+ keys is done by default. The **0** parameter disables encryption. The **1** parameter is not required; it is provided for backwards compatibility.

Setting the retransmission limit

The **retransmit** parameter specifies how many times the Ruckus device will resend an authentication request when the TACACS/TACACS+ server does not respond. The retransmit limit can be from 1 - 5 times. The default is 3 times.

To set the TACACS and TACACS+ retransmit limit, enter a command such as the following.

```
device(config)#tacacs-server retransmit 5
```

Setting the timeout parameter

The **timeout** parameter specifies how many seconds the Ruckus device waits for a response from the TACACS/TACACS+ server before either retrying the authentication request, or determining that the TACACS/TACACS+ server is unavailable and moving on to the next authentication method in the authentication-method list. The timeout can be from 1 - 15 seconds. The default is 3 seconds.

```
device(config)#tacacs-server timeout 5
```

Enabling management access based on a port-based VLAN

You can restrict management access so that only devices with ports in a specific port-based VLAN have access. Clients connected to ports that are not in the VLAN are denied management access. VLAN-based access control works in conjunction with other access control methods.

The following considerations apply to port-based VLAN access control.

- As in a switched network, the TACACS server and the SSH client should be in the same VLAN.
- Otherwise, the response expected from the TACACS server should be sent in the same VLAN as configured by the **tacacs-server enable vlan** command. With this configuration, the TACACS server can be in a different VLAN and still allow SSH connections in a routed network.
- The **tacacs-server enable vlan** command should not be configured in a network that uses dynamic routing because the TACACS server response might be routed on any path.

The following example allows TACACS server management access only to clients in VLAN 10.

```
device# configure terminal
device(config)# tacacs-server enable vlan 10
```

Configuring authentication-method lists forTACACS and TACACS+

You can use TACACS/TACACS+ to authenticate Telnet/SSH access and access to Privileged EXEC level and CONFIG levels of the CLI. When configuring TACACS/TACACS+ authentication, you create authentication-method lists specifically for these access methods, specifying TACACS/TACACS+ as the primary authentication method.

Within the authentication-method list, TACACS/TACACS+ is specified as the primary authentication method and up to six backup authentication methods are specified as alternates. If TACACS/TACACS+ authentication fails due to an error, the device tries the backup authentication methods in the order they appear in the list.

When you configure authentication-method lists for TACACS/TACACS+ authentication, you must create a separate authentication-method list for Telnet/SSH CLI access, and for access to the Privileged EXEC level and CONFIG levels of the CLI.

To create an authentication method list that specifies TACACS/TACACS+ as the primary authentication method for securing Telnet/SSH access to the CLI.

```
device(config)#enable telnet authentication
device(config)#aaa authentication login default tacacs local
```

The commands above cause TACACS/TACACS+ to be the primary authentication method for securing Telnet/SSH access to the CLI. If TACACS/TACACS+ authentication fails due to an error with the server, authentication is performed using local user accounts instead.

To create an authentication-method list that specifies TACACS/TACACS+ as the primary authentication method for securing access to Privileged EXEC level and CONFIG levels of the CLI.

```
device(config)#aaa authentication enable default tacacs local none
```

The command above causes TACACS/TACACS+ to be the primary authentication method for securing access to Privileged EXEC level and CONFIG levels of the CLI. If TACACS/TACACS+ authentication fails due to an error with the server, local authentication is used instead. If local authentication fails, no authentication is used; the device automatically permits access.

The **web-server** | **enable** | **login** parameter specifies the type of access this authentication-method list controls. You can configure one authentication-method list for each type of access.

NOTE
If you configure authentication for Web management access, authentication is performed each time a page is requested from the server. When frames are enabled on the Web Management Interface, the browser sends an HTTP request for each frame. The ICX device authenticates each HTTP request from the browser. To limit authentications to one per page, disable frames on the Web Management Interface.

The *method1* parameter specifies the primary authentication method. The remaining optional *method* parameters specify additional methods to try if an error occurs with the primary method. A method can be one of the values listed in the Method Parameter column in the following table.

TABLE 4 Authentication method values

Method parameter	Description
line	Authenticate using the password you configured for Telnet access. The Telnet password is configured using the **enable telnet password...** command. Refer to the Setting a telnet password task.
enable	Authenticate using the password you configured for the Super User privilege level. This password is configured using the **enable super-user-password...** command. Refer to the Setting passwords for management privilege levels task.
local	Authenticate using a local user name and password you configured on the device. Local user names and passwords are configured using the **username...** command. Refer to the Local user account configuration task.
tacacs	Authenticate using the database on a TACACS server. You also must identify the server to the device using the **tacacs-server** command.
tacacs+	Authenticate using the database on a TACACS+ server. You also must identify the server to the device using the **tacacs-server** command.
radius	Authenticate using the database on a RADIUS server. You also must identify the server to the device using the **radius-server** command.
none	Do not use any authentication method. The device automatically permits access.

Entering privileged EXEC mode after a Telnet or SSH login

By default, a user enters User EXEC mode after a successful login through Telnet or SSH. Optionally, you can configure the device so that a user enters Privileged EXEC mode after a Telnet or SSH login. To do this, use the following command.

```
device(config)#aaa authentication login privilege-mode
```

The user privilege level is based on the privilege level granted during login.

Configuring enable authentication to prompt for password only

If Enable authentication is configured on the device, when a user attempts to gain Super User access to the Privileged EXEC and CONFIG levels of the CLI, by default he or she is prompted for a username and password. You can configure the Ruckus device to prompt only for a password. The device uses the username entered at login, if one is available. If no username was entered at login, the device prompts for both username and password.

To configure the Ruckus device to prompt only for a password when a user attempts to gain Super User access to the Privileged EXEC and CONFIG levels of the CLI.

```
device(config)#aaa authentication enable implicit-user
```

Telnet and SSH prompts when the TACACS+ Server is unavailable

When TACACS+ is the first method in the authentication method list, the device displays the login prompt received from the TACACS+ server. If a user attempts to login through Telnet or SSH, but none of the configured TACACS+ servers are available, the following takes place:

- If the next method in the authentication method list is "enable", the login prompt is skipped, and the user is prompted for the Enable password (that is, the password configured with the **enable super-user-password** command).
- If the next method in the authentication method list is "line", the login prompt is skipped, and the user is prompted for the Line password (that is, the password configured with the **enable telnet password** command).

Configuring TACACS+ authorization

Ruckus devices support TACACS+ authorization for controlling access to management functions in the CLI. Two kinds of TACACS+ authorization are supported:

- Exec authorization determines a user privilege level when they are authenticated
- Command authorization consults a TACACS+ server to get authorization for commands entered by the user

Configuring exec authorization

When TACACS+ exec authorization is performed, the Ruckus device consults a TACACS+ server to determine the privilege level of the authenticated user. To configure TACACS+ exec authorization on the Ruckus device, enter the following command.

```
device(config)#aaa authorization exec default tacacs+
```

If you specify **none** , or omit the **aaa authorization exec** command from the device configuration, no exec authorization is performed.

A user privilege level is obtained from the TACACS+ server in the "foundry-privlvl" A-V pair. If the **aaa authorization exec default tacacs+** command exists in the configuration, the device assigns the user the privilege level specified by this A-V pair. If the command does not exist in the configuration, then the value in the "foundry-privlvl" A-V pair is ignored, and the user is granted Super User access.

NOTE
The TACACS+ server is a separate device made by third-party manufacturers. It is used to authenticate clients logging into the Ruckus device using telnet, SSH, or console. There are multiple applications available for configuring TACACS+ server, such as tac_plus for Linux and Cisco ACS for Windows. Ruckus recommends setting the client user's "privlvl" attribute to 15, because this assigns super-user privileges to the authenticated client. If the "privlvl" attribute is not available under the user configuration options for your TACACS+ server software, please refer to the TACACS+ server's support documentation.

NOTE
If the **aaa authorization exec default tacacs+** command exists in the configuration, following successful authentication the device assigns the user the privilege level specified by the "foundry-privlvl" A-V pair received from the TACACS+ server. If the **aaa authorization exec default tacacs+** command does not exist in the configuration, then the value in the "foundry-privlvl" A-V pair is ignored, and the user is granted Super User access.Also note that in order for the **aaa authorization exec default tacacs+** command to work, either the**aaa authentication enable default tacacs +** command, or the **aaa authentication login privilege-mode** command must also exist in the configuration.

Configuring an Attribute-Value pair on the TACACS+ server

During TACACS+ exec authorization, the Ruckus device expects the TACACS+ server to send a response containing an A-V (Attribute-Value) pair that specifies the privilege level of the user. When the Ruckus device receives the response, it extracts an A-V pair configured for the Exec service and uses it to determine the user privilege level.

To set a user privilege level, you can configure the "foundry-privlvl" A-V pair for the Exec service on the TACACS+ server.

```
user=bob {
   default service = permit
   member admin
   #Global password
   global = cleartext "cat"
   service = exec {
     foundry-privlvl = 0
        }
}
```

In this example, the A-V pair `foundry-privlvl = 0` grants the user full read-write access. The value in the foundry-privlvl A-V pair is an integer that indicates the privilege level of the user. Possible values are 0 for super-user level, 4 for port-config level, or 5 for read-only level. If a value other than 0, 4, or 5 is specified in the foundry-privlvl A-V pair, the default privilege level of 5 (read-only) is used. The foundry-privlvl A-V pair can also be embedded in the group configuration for the user. See your TACACS+ documentation for the configuration syntax relevant to your server.

If the foundry-privlvl A-V pair is not present, the Ruckus device extracts the last A-V pair configured for the Exec service that has a numeric value. The Ruckus device uses this A-V pair to determine the user privilege level.

```
user=bob {
   default service = permit
   member admin
   #Global password
   global = cleartext "cat"
   service = exec {
     privlvl = 15
        }
}
```

The attribute name in the A-V pair is not significant; the Ruckus device uses the last one that has a numeric value. However, the Ruckus device interprets the value for a non-"foundry-privlvl" A-V pair differently than it does for a "foundry-privlvl" A-V pair. The following table lists how the Ruckus device associates a value from a non-"foundry-privlvl" A-V pair with a Ruckus privilege level.

TABLE 5 Ruckus equivalents for non-"foundry-privlvl" A-V pair values

Value for non-"foundry-privlvl" A-V pair	Ruckus privilege level
15	0 (super-user)
From 14 - 1	4 (port-config)
Any other number or 0	5 (read-only)

In the example above, the A-V pair configured for the Exec service is `privlvl = 15` . The Ruckus device uses the value in this A-V pair to set the user privilege level to 0 (super-user), granting the user full read-write access.

In a configuration that has both a "foundry-privlvl" A-V pair and a non-"foundry-privlvl" A-V pair for the Exec service, the non-"foundry-privlvl" A-V pair is ignored.

```
user=bob {
   default service = permit
   member admin
   #Global password
   global = cleartext "cat"
   service = exec {
     foundry-privlvl = 4
     privlvl = 15
        }
}
```

In this example, the user would be granted a privilege level of 4 (port-config level). The `privlvl = 15` A-V pair is ignored by the Ruckus device.

If the TACACS+ server has no A-V pair configured for the Exec service, the default privilege level of 5 (read-only) is used.

Configuring command authorization

When TACACS+ command authorization is enabled, the Ruckus device consults a TACACS+ server to get authorization for commands entered by the user.

You enable TACACS+ command authorization by specifying a privilege level whose commands require authorization. For example, to configure the Ruckus device to perform authorization for the commands available at the Super User privilege level (that is, all commands on the device), enter the following command.

```
device(config)#aaa authorization commands 0 default tacacs+
```

The privilege-level parameter can be one of the following:

- **0** - Authorization is performed for commands available at the Super User level (all commands)
- **4** - Authorization is performed for commands available at the Port Configuration level (port-config and read-only commands)
- **5** - Authorization is performed for commands available at the Read Only level (read-only commands)

  NOTE
  TACACS+ command authorization can be performed only for commands entered from Telnet or SSH sessions, or from the console. No authorization is performed for commands entered at the Web Management Interface.

TACACS+ command authorization is not performed for the following commands:

- At all levels: **exit** , **logout** , **end** , and **quit** .
- At the Privileged EXEC level: **enable** or **enable** text , where text is the password configured for the Super User privilege level.

If configured, command accounting is performed for these commands.

AAA support for console commands

AAA support for commands entered at the console includes the following:

- Login prompt that uses AAA authentication, using authentication-method Lists
- Exec Authorization
- Exec Accounting
- Command authorization
- Command accounting
- System Accounting

To enable AAA support for commands entered at the console, enter the following command.

```
device(config)#enable aaa console
```

TACACS+ accounting configuration

Ruckus devices support TACACS+ accounting for recording information about user activity and system events. When you configure TACACS+ accounting on a Ruckus device, information is sent to a TACACS+ accounting server when specified events occur, such as when a user logs into the device or the system is rebooted.

Configuring TACACS+ accounting for Telnet/SSH (Shell) access

To send an Accounting Start packet to the TACACS+ accounting server when an authenticated user establishes a Telnet or SSH session on the Ruckus device, and an Accounting Stop packet when the user logs out.

```
device(config)#aaa accounting exec default start-stop tacacs+
```

Configuring TACACS+ accounting for CLI commands

You can configure TACACS+ accounting for CLI commands by specifying a privilege level whose commands require accounting. For example, to configure the Ruckus device to perform TACACS+ accounting for the commands available at the Super User privilege level (that is; all commands on the device), enter the following command.

```
device(config)#aaa accounting commands 0 default start-stop tacacs+
```

An Accounting Start packet is sent to the TACACS+ accounting server when a user enters a command, and an Accounting Stop packet is sent when the service provided by the command is completed.

> **NOTE**
> If authorization is enabled, and the command requires authorization, then authorization is performed before accounting takes place. If authorization fails for the command, no accounting takes place.

The *privilege-level* parameter can be one of the following:

- **0** - Records commands available at the Super User level (all commands)
- **4** - Records commands available at the Port Configuration level (port-config and read-only commands)
- **5** - Records commands available at the Read Only level (read-only commands)

Configuring TACACS+ accounting for system events

You can configure TACACS+ accounting to record when system events occur on the Ruckus device. System events include rebooting and when changes to the active configuration are made.

The following command causes an Accounting Start packet to be sent to the TACACS+ accounting server when a system event occurs, and a Accounting Stop packet to be sent when the system event is completed.

```
device(config)#aaa accounting system default start-stop tacacs+
```

Configuring an interface as the source for all TACACS and TACACS+ packets

You can designate the lowest-numbered IP address configured an Ethernet port, loopback interface, or virtual interface as the source IP address for all TACACS/TACACS+ packets from the Layer 3 device.

Displaying TACACS/TACACS+ statistics and configuration information

The **show aaa** command displays information about all TACACS+ and RADIUS servers identified on the device.

```
device#show running-config aaa
Tacacs+ key: foundry
Tacacs+ retries: 1
Tacacs+ timeout: 15 seconds
Tacacs+ dead-time: 3 minutes
Tacacs+ Server: 10.95.6.90 Port:49:
                opens=6 closes=3 timeouts=3 errors=0
                packets in=4 packets out=4
no connection
Radius key: networks
Radius retries: 3
Radius timeout: 3 seconds
Radius dead-time: 3 minutes
Radius Server: 10.95.6.90 Auth Port=1812 Acct Port=1813:
                opens=2 closes=1 timeouts=1 errors=0
                packets in=1 packets out=4
no connection
```

The following table describes the TACACS/TACACS+ information displayed by the **show aaa** command.

TABLE 6 Output of the show aaa command for TACACS/TACACS+

Field	Description
Tacacs+ key	The setting configured with the **tacacs-server key** command. At the Super User privilege level, the actual text of the key is displayed. At the other privilege levels, a string of periods (....) is displayed instead of the text.
Tacacs+ retries	The setting configured with the **tacacs-server retransmit** command.
Tacacs+ timeout	The setting configured with the **tacacs-server timeout** command.
Tacacs+ dead-time	The setting configured with the **tacacs-server dead-time** command.
Tacacs+ Server	For each TACACS/TACACS+ server, the IP address, port, and the following statistics are displayed: • opens - Number of times the port was opened for communication with the server • closes - Number of times the port was closed normally • timeouts - Number of times port was closed due to a timeout • errors - Number of times an error occurred while opening the port • packets in - Number of packets received from the server

TABLE 6 Output of the show aaa command for TACACS/TACACS+ (continued)

Field	Description
	• packets out - Number of packets sent to the server
connection	The current connection status. This can be "no connection" or "connection active".

The **show web connection** command displays the privilege level of Web Management Interface users.

Example

```
device# show web-connection
We management Sessions:
User Privilege IP address MAC address Timeout(secs) Connection
roy READ-WRITE 10.1.1.3 0030.488.b84d9 279 HTTPS
```

Use the following command to clear web connections:

```
device# clear web-connection
```

After issuing the **clear web connection** command, the **show web connection** command displays the following output:

```
device# show web-connection
No WEB-MANAGEMENT sessions are currently established!
```

RADIUS Authentication

RADIUS security

You can use a Remote Authentication Dial In User Service (RADIUS) server to secure the following types of access to the Ruckus Layer 2 Switch or Layer 3 Switch:

- Telnet access
- SSH access
- Web management access
- TLS support
- Access to the Privileged EXEC level and CONFIG levels of the CLI

When RADIUS authentication is implemented, the Ruckus device consults a RADIUS server to verify user names and passwords. You can optionally configure RADIUS authorization, in which the Ruckus device consults a list of commands supplied by the RADIUS server to determine whether a user can issue the entered command.

FastIron supports TLS encryption for RADIUS server and client authentication.

RADIUS accounting causes the Ruckus device to log information on a RADIUS accounting server when specified events occur on the device.

RADIUS authentication

When RADIUS authentication takes place, the following events occur.

1. A user attempts to gain access to the Ruckus device by doing one of the following:
   - Logging into the device using Telnet, SSH, or the Web Management Interface
   - Entering the Privileged EXEC level or CONFIG level of the CLI
2. The user is prompted for a username and password.
3. The user enters a username and password.
4. The Ruckus device sends a RADIUS Access-Request packet containing the username and password to the RADIUS server.
5. The RADIUS server validates the Ruckus device using a shared secret (the RADIUS key).
6. The RADIUS server looks up the username in its database.
7. If the username is found in the database, the RADIUS server validates the password.
8. If the password is valid, the RADIUS server sends an Access-Accept packet to the Ruckus device, authenticating the user. Within the Access-Accept packet are three Ruckus vendor-specific attributes that indicate the following:
   - The privilege level of the user
   - A list of commands
   - Whether the user is allowed or denied usage of the commands in the list

   The last two attributes are used with RADIUS authorization, if configured.
9. The user is authenticated, and the information supplied in the Access-Accept packet for the user is stored on the Ruckus device. The user is granted the specified privilege level. If you configure RADIUS authorization, the user is allowed or denied usage of the commands in the list.

RADIUS authorization

When RADIUS authorization takes place, the following events occur.

1. A user previously authenticated by a RADIUS server enters a command on the Ruckus device.
2. The Ruckus device looks at its configuration to see if the command is at a privilege level that requires RADIUS command authorization.
3. If the command belongs to a privilege level that requires authorization, the Ruckus device looks at the list of commands delivered to it in the RADIUS Access-Accept packet when the user was authenticated. (Along with the command list, an attribute was sent that specifies whether the user is permitted or denied usage of the commands in the list.)

   NOTE
   After RADIUS authentication takes place, the command list resides on the Ruckus device. The RADIUS server is not consulted again once the user has been authenticated. This means that any changes made to the user command list on the RADIUS server are not reflected until the next time the user is authenticated by the RADIUS server, and the new command list is sent to the Ruckus device.

4. If the command list indicates that the user is authorized to use the command, the command is executed.

RADIUS accounting

RADIUS accounting works as follows.

1. One of the following events occurs on the Ruckus device:
   - A user logs into the management interface using Telnet or SSH
   - A user enters a command for which accounting has been configured
   - A system event occurs, such as a reboot or reloading of the configuration file
2. The Ruckus device checks its configuration to see if the event is one for which RADIUS accounting is required.
3. If the event requires RADIUS accounting, the Ruckus device sends a RADIUS Accounting Start packet to the RADIUS accounting server, containing information about the event.
4. The RADIUS accounting server acknowledges the Accounting Start packet.
5. The RADIUS accounting server records information about the event.
6. When the event is concluded, the Ruckus device sends an Accounting Stop packet to the RADIUS accounting server.
7. The RADIUS accounting server acknowledges the Accounting Stop packet.

AAA operations for RADIUS

The following table lists the sequence of authentication, authorization, and accounting (AAA) operations that take place when a user gains access to a Ruckus device that has RADIUS security configured. For each action the user takes, there are certain applicable AAA operations that occur. This table is presented as a means to troubleshoot possible issues with AAA configuration.

User action	Applicable AAA operations
User attempts to gain access to the Privileged EXEC and CONFIG levels of the CLI	Enable authentication: aaa authentication enable default method-list
	System accounting start: aaa accounting system default start-stop method-list
User logs in using Telnet/SSH	Login authentication: aaa authentication login default method-list
	EXEC accounting Start: aaa accounting exec default start-stop method-list
	System accounting Start: aaa accounting system default start-stop method-list
User logs into the Web Management Interface	Web authentication: aaa authentication web-server default <method-list>
User logs out of Telnet/SSH session	Command authorization for **logout** command: aaa authorization commands privilege-level default method-list
	Command accounting: aaa accounting commands privilege-level default start-stop method-list
	EXEC accounting stop: aaa accounting exec default start-stop method-list
User enters system commands (for example, **reload**, **boot system**)	Command authorization: aaa authorization commands privilege -level default method-list
	Command accounting: aaa accounting commands privilege-level default start-stop method-list
	System accounting stop: aaa accounting system default start-stop method-list

User action	Applicable AAA operations
User enters the command: **[no] aaa accounting system default start-stop method-list**	Command authorization: aaa authorization commands privilege-level default method-list
	Command accounting: aaa accounting commands privilege-level default start-stop method-list
	System accounting start: aaa accounting system default start-stop method-list
User enters other commands	Command authorization: aaa authorization commands privilege-level default method-list
	Command accounting: aaa accounting commands privilege-level default start-stop method-list

AAA security for commands pasted Into the running-config

If AAA security is enabled on the device, commands pasted into the running-config are subject to the same AAA operations as if they were entered manually.

When you paste commands into the running-config, and AAA command authorization or accounting, or both, are configured on the device, AAA operations are performed on the pasted commands. The AAA operations are performed before the commands are actually added to the running-config. The server performing the AAA operations should be reachable when you paste the commands into the running-config file. If the device determines that a pasted command is invalid, AAA operations are halted on the remaining commands. The remaining commands may not be issued if command authorization is configured.

NOTE
Since RADIUS command authorization relies on a list of commands received from the RADIUS server when authentication is performed, it is important that you use RADIUS authentication when you also use RADIUS command authorization.

RADIUS configuration considerations

If you are using RADIUS authentication, authorization, and accounting, be aware of the following considerations:

- You must deploy at least one RADIUS server in your network in order for the RADIUS feature to function.
- Ruckus devices support authentication using up to eight RADIUS servers, including those used for 802.1X authentication and for management. The device tries to use the servers in the order you add them to the device configuration. If one RADIUS server times out (does not respond), the Ruckus device tries the next one in the list. Servers are tried in the same sequence each time there is a request.
- You can optionally configure a RADIUS server as a port server, indicating that the server will be used only to authenticate users on ports to which it is mapped, as opposed to globally authenticating users on all ports of the device. Refer to RADIUS server per port on page 62.
- You can map up to eight RADIUS port servers to each port on the Ruckus device. The port will authenticate users using only the RADIUS servers to which it is mapped. If there are no RADIUS servers mapped to a port, it will use the "global" servers for authentication. Refer to RADIUS server to individual ports mapping on page 62.
- You can select only one primary authentication method for each type of access to a device (CLI through Telnet, CLI Privileged EXEC and CONFIG levels). For example, you can select RADIUS as the primary authentication method for

Telnet CLI access, but you cannot also select TACACS+ authentication as the primary method for the same type of access. However, you can configure different backup authentication methods for each access type.

- If your Ruckus device is configured with multiple IP addresses, the lowest numbered IP sources the device's RADIUS traffic by default. You may select a specific interface by configuring the IP RADIUS SOURCE-INTERFACE {interface-type +ID}. If the selected interface has multiple IPs, the lowest-numbered is used.

Configuring RADIUS

Follow the procedure given below to configure a Ruckus device for RADIUS.

1. Configure Ruckus vendor-specific attributes on the RADIUS server. Refer to Company-specific attributes on the RADIUS server on page 59.
2. Identify the RADIUS server to the Ruckus device. Refer to Identifying the RADIUS server to the Ruckus device on page 61.
3. Optionally specify different servers for individual AAA functions. Refer to Specifying different servers for individual AAA functions on page 61.
4. Optionally configure the RADIUS server as a "port only" server. Refer to RADIUS server per port on page 62.
5. Optionally bind the RADIUS servers to ports on the Ruckus device. Refer to RADIUS server to individual ports mapping on page 62.
6. Set RADIUS parameters. Refer to RADIUS parameters on page 63.
7. Configure authentication-method lists. Refer to Setting authentication-method lists for RADIUS on page 64.
8. Optionally configure RADIUS authorization. Refer to RADIUS authorization on page 66.
9. Optionally configure RADIUS accounting. Refer to RADIUS accounting on page 68.

Company-specific attributes on the RADIUS server

NOTE
For all Ruckus devices, RADIUS Challenge is supported for 802.1x authentication but not for login authentication.

During the RADIUS authentication process, if a user supplies a valid username and password, the RADIUS server sends an Access-Accept packet to the Ruckus device, authenticating the user. Within the Access-Accept packet are three required Ruckus vendor-specific attributes that indicate the following:

- The privilege level of the user
- A list of commands
- Whether the user is allowed or denied usage of the commands in the list

You must add at least these three Ruckus vendor-specific attributes to your RADIUS server configuration, and configure the attributes in the individual or group profiles of the users that will access the Ruckus device.

Ruckus Vendor-ID is 1991, with Vendor-Type 1.

The following table describes the all of the available Ruckus vendor-specific attributes.

TABLE 7 Ruckus vendor-specific attributes for RADIUS

Attribute name	Attribute ID	Data type	Description
foundry-privilege-level	1	integer	Specifies the privilege level for the user. This attribute can be set to one of the following: • **0** - **Super User level** - Allows complete read-and-write access to the system. This is generally for system administrators and is the only management privilege level that allows you to configure passwords. • **4** - **Port Configuration level** - Allows read-and-write access for specific ports but not for global (system-wide) parameters. • **5** - **Read Only level** - Allows access to the Privileged EXEC mode and User EXEC mode of the CLI but only with read access.
foundry-command-string	2	string	Specifies a list of CLI commands that are permitted or denied to the user when RADIUS authorization is configured. The commands are delimited by semi-colons (;). You can specify an asterisk (*) as a wildcard at the end of a command string. For example, the following command list specifies all **show** and **debug ip** commands, as well as the **write terminal** command: show *; debug ip *; write term*
foundry-command-exception-flag	3	integer	Specifies whether the commands indicated by the foundry-command-string attribute are permitted or denied to the user. This attribute can be set to one of the following: • **0** - Permit execution of the commands indicated by foundry-command-string, deny all other commands. • **1** - Deny execution of the commands indicated by foundry-command-string, permit all other commands.
foundry-access-list	5	string	Specifies the access control list to be used for RADIUS authorization. Enter the access control list in the following format. type=string, value="ipacl.[e|s].[in|out] = [acl-name | acl-number] separator macfilter.in = [acl-name | acl-number] Where: • separator can be a space, newline, semicolon, comma, or null characater • ipacl.e is an extended ACL; ipacl.s is a standard ACL.
foundry-MAC-authent-needs-802x	6	integer	Specifies whether or not 802.1x authentication is required and enabled. **0** - Disabled **1** - Enabled
foundry-802.1x-valid-lookup	7	integer	Specifies if 802.1x lookup is enabled: **0** - Disabled **1** - Enabled
foundry-MAC-based-VLAN-QOS	8	integer	Specifies the priority for MAC-based VLAN QOS: **0** - qos_priority_0 **1** - qos_priority_1 **2** - qos_priority_2 **3** - qos_priority_3

TABLE 7 Ruckus vendor-specific attributes for RADIUS (continued)

Attribute name	Attribute ID	Data type	Description
			4 - qos_priority_4 **5** - qos_priority_5 **6** - qos_priority_6 **7** - qos_priority_7
foundry-coa-command	10	string	Specifies to perform CoA command dynamically on the port or host after the device or user is authenticated. **disable-port** - Disables the specified port. **reauth-host** - Re-authenticate the host specified by MAC address. **flip-port** - Brings the port up and down with some delay between the toggle. **modify-acl** - Replace the specified ACL with the session's existing ACL. Modify-ACL is supported with the Filter-Id (11) attribute. The IP ACL specified through the Filter-Id attribute replaces the session's existing ACL configuration.

Identifying the RADIUS server to the Ruckus device

To use a RADIUS server to authenticate access to a Ruckus device, you must identify the server to the Ruckus device using the **radius-server host** command as shown in the following example.

```
device# configure terminal
device(config)# radius-server host 10.157.22.99
```

The example configures the RADIUS server with the IP address 10.157.22.99 as the authentication server.

Specifying different servers for individual AAA functions

Separate RADIUS servers can be configured and assigned for specific AAA tasks. For example, you can designate one RADIUS server to handle authentication and another RADIUS server to handle accounting. You can specify individual servers for authentication and accounting, but not for authorization. You can set the RADIUS key for each server.

To specify different RADIUS servers for authentication and accounting, enter commands such as the following.

```
device(config)# radius-server host 10.2.3.4 authentication-only key abc
device(config)# radius-server host 10.2.3.6 accounting-only key ghi
```

TLS and RADIUS

The **ssl-auth-port** keyword specifies that the server is a RADIUS server running over a TLS-encrypted TCP session. Only one auth-port or ssl-auth-port can be specified. If neither is specified, it defaults to existing default behavior, which is to use the default auth-port of 1812 and 1813 for accounting with no TLS encryption. TLS-encrypted sessions support both IPv4 and IPv6.

NOTE
TLS-encrypted TCP sessions are not supported by management VRF.

After authentication takes place, the server that performed the authentication is used for authorization and accounting. If the authenticating server cannot perform the requested function, then the next server in the configured list of servers is tried; this process repeats until a server that can perform the requested function is found, or every server in the configured list has been tried.

RADIUS server per port

You can optionally configure a RADIUS server per port, indicating that it will be used only to authenticate users on ports to which it is mapped. A RADIUS server that is not explicitly configured as a RADIUS server per port is a global server, and can be used to authenticate users on ports to which no RADIUS servers are mapped.

RADIUS server per port configuration notes

- This feature works with 802.1X and MAC authentication only.
- You can define up to eight RADIUS servers per Ruckus device.

RADIUS configuration example and command syntax

The following shows an example RADIUS host configuration.

```
device(config)# radius-server host 10.10.10.103 auth-port 1812 acct-port 1813 default key mykeyword dot1x
port-only
device(config)# radius-server host 10.10.10.104 auth-port 1812 acct-port 1813 default key mykeyword dot1x
port-only
device(config)# radius-server host 10.10.10.105 auth-port 1812 acct-port 1813 default key mykeyword dot1x
device(config)# radius-server host 10.10.10.106 auth-port 1812 acct-port 1813 default key mykeyword dot1x
device(config)# radius-server host 10.10.10.106 auth-port 1812 acct-port 1813 default key mykeyword mac-auth
device(config)# radius-server host 10.10.10.106 auth-port 1812 acct-port 1813 default key mykeyword web-auth
```

The configuration has the following effect:

- RADIUS servers 10.10.10.103 and 10.10.10.104 are used only to authenticate users on ports to which the servers are mapped. To map a RADIUS server to a port, refer to RADIUS server to individual ports mapping on page 62.
- RADIUS servers 10.10.10.105 and 10.10.10.106 are used to authenticate users on ports to which no RADIUS servers are mapped. For example, port e 9, to which no RADIUS servers are mapped, sends a RADIUS request to the first configured RADIUS server, 10.10.10.105. If the request fails, it goes to the second configured RADIUS server, 10.10.10.106. It does not send requests to 10.10.10.103 or 10.10.10.104, since these servers are configured as port servers.

RADIUS server to individual ports mapping

You can map up to eight RADIUS servers to each port on the Ruckus device. The port will authenticate users using only the RADIUS servers to which the port is mapped. If there are no RADIUS servers mapped to a port, it will use the "global" servers for authentication.

As in previous releases, a port goes through the list of servers in the order in which it was mapped or configured, until a server that can perform the requested function is found, or until every server in the list has been tried.

RADIUS server-to-ports configuration notes

- This feature works with 802.1X and MAC authentication only.
- You can map a RADIUS server to a physical port only. You cannot map a RADIUS server to a VE.

RADIUS server-to-ports configuration example and command syntax

The following example maps a RADIUS server to a specific port.

```
device# configure terminal
device(config)# dot1x port-control auto ethernet 3/1/1
device(config)# interface ethernet 3/1/1
device(config-if-e1000-3/1/1)# use-radius-server 10.10.10.103
device(config-if-e1000-3/1/1)# use-radius-server 10.10.10.110
```

The example uses the **dot1x port-control** command to authorize port 3/1/1 once the client has been authenticated on an 802.1x-enabled interface. The example configures two RADIUS servers for port 3/1/1 to use. The port first sends a RADIUS request to the RADIUS server with IP address 10.10.10.103 because it is the first server mapped to the port. If the request fails, the port sends a request to the RADIUS server at IP address 10.10.10.110.

RADIUS parameters

You can set the following parameters in a RADIUS configuration:

- RADIUS key - This parameter specifies the value that the Ruckus device sends to the RADIUS server when trying to authenticate user access.
- Retransmit interval - This parameter specifies how many times the Ruckus device will resend an authentication request when the RADIUS server does not respond. The retransmit value can be from 1 - 5 times. The default is 3 times.
- Timeout - This parameter specifies how many seconds the Ruckus device waits for a response from a RADIUS server before either retrying the authentication request, or determining that the RADIUS servers are unavailable and moving on to the next authentication method in the authentication-method list. The timeout can be from 1 - 15 seconds. The default is 3 seconds.

Setting the RADIUS key

The **key** parameter in the **radius-server** command is used to encrypt RADIUS packets before they are sent over the network. The value for the **key** parameter on the Ruckus device should match the one configured on the RADIUS server. The key can be from 1 - 64 characters in length and cannot include any space characters.

To specify a RADIUS server key, enter a command such as the following.

```
device# configure terminal
device(config)# radius-server key mirabeau
```

When you display the configuration of the Ruckus device, the RADIUS key is encrypted.

```
device# configure terminal
device(config)# radius-server key abc
device(config)# write terminal
...
device(config)# show running-config | in radius
radius-server key abc
```

Setting the RADIUS retransmission limit

The **retransmit** parameter specifies the maximum number of retransmission attempts. When an authentication request times out, the Ruckus device retransmits the request up to the maximum number of retransmissions configured. The default retransmit value is 2 retries.

To set the RADIUS retransmit limit, enter the **radius-server retransmit** command followed by a value from 1 through 5 as shown in the following example.

```
device# configure terminal
device(config)# radius-server retransmit 5
```

The example configures the Ruckus device to retransmit a RADIUS authentication request no more than 5 times.

Setting the timeout parameter

The **timeout** parameter specifies how many seconds the Ruckus device waits for a response from the RADIUS server before either retrying the authentication request, or determining that the RADIUS server is unavailable and moving on to the next authentication method in the authentication-method list.

To specify the RADIUS timeout, enter the **radius-server timeout** command followed by a value in seconds as shown in the following example.

```
device# configure terminal
device(config)# radius-server timeout 5
```

The timeout can be from 1 through 15 seconds. The default is 3 seconds.

Setting RADIUS over IPv6

Ruckus devices can send RADIUS packets over an IPv6 network.

To enable the Ruckus device to send RADIUS packets over IPv6, enter the **radius-server host** command followed by the keyword **ipv6** and the IPv6 address of the host as shown in the following example.

NOTE
When you enter the IPv6 host address, you do not need to specify the prefix length. A prefix length of 128 is implied.

```
device# configure terminal
device(config)# radius-server host ipv6 2001:DB8::300
```

The example configures the RADIUS server with the IPv6 address 2001:DB8::300.

Setting authentication-method lists for RADIUS

You can use RADIUS to authenticate Telnet/SSH access and access to Privileged EXEC and CONFIG levels of the CLI. When configuring RADIUS authentication, you create authentication-method lists specifically for these access methods, specifying RADIUS as the primary authentication method.

Within the authentication-method list, RADIUS is specified as the primary authentication method and up to six other authentication methods are specified as alternates. If RADIUS authentication fails, the device tries the alternate authentication methods in the order they appear in the list.

When you configure authentication-method lists for RADIUS, you must create a separate authentication-method list for Telnet or SSH CLI access and for CLI access to the Privileged EXEC and CONFIG levels of the CLI.

The following example creates an authentication-method list that first lists the primary authentication method for securing Telnet access to the CLI (in this case, RADIUS).

```
device# configure terminal
device(config)# enable telnet authentication
device(config)# aaa authentication login default radius local
```

The example configures RADIUS as the primary authentication method for securing Telnet access to the CLI. If RADIUS authentication fails due to a server error, local authentication is used instead.

The following example creates an authentication-method list that configures authentication methods for securing access to Privileged EXEC level and CONFIG levels of the CLI.

```
device# configure terminal
device(config)# aaa authentication enable default radius local none
```

The example configures RADIUS as the primary authentication method for securing access to Privileged EXEC and CONFIG levels of the CLI. If RADIUS authentication fails due to a server error, local authentication is used instead. If local authentication fails, no authentication is used, and the device automatically permits access.

NOTE
If you configure authentication for Web management access, authentication is performed each time a page is requested from the server. When frames are enabled on the Web Management Interface, the browser sends an HTTP request for each frame. The ICX device authenticates each HTTP request from the browser. To limit authentications to one per page, disable frames on the Web Management Interface.

Available authentication methods are listed in the following table.

TABLE 8 Authentication method values

Method parameter	Description
line	Authenticate using the password you configured for Telnet access. The Telnet password is configured using the **enable telnet password...** command. Refer to Setting a Telnet password on page 19.
enable	Authenticate using the password you configured for the Super User privilege level. This password is configured using the **enable super-user-password...** command. Refer to Setting passwords for management privilege levels on page 19.
local	Authenticate using a local user name and password you configured on the device. Local user names and passwords are configured using the **username...** command. Refer to Local user account configuration on page 26.
tacacs	Authenticate using the database on a TACACS server. You also must identify the server to the device using the **tacacs-server** command.
tacacs+	Authenticate using the database on a TACACS+ server. You also must identify the server to the device using the **tacacs-server** command.
radius	Authenticate using the database on a RADIUS server. You also must identify the server to the device using the **radius-server** command.
none	Do not use any authentication method. The device automatically permits access.

NOTE
For examples of how to define authentication-method lists for types of authentication other than RADIUS, refer to Authentication-method lists on page 77.

Entering privileged EXEC mode after a Telnet or SSH login

The user privilege level is based on the privilege level granted during login. By default, a user enters User EXEC mode after a successful login through Telnet or SSH. You have the option to configure the device so that a user enters Privileged EXEC mode after a Telnet or SSH login. To do this, enter the following command.

```
device# configure terminal
device(config)# aaa authentication login privilege-mode
```

Configuring enable authentication to prompt for password only

If **enable authentication** is configured on the device, by default, the user is prompted for a username and password when attempting to gain Super User access to the Privileged EXEC and CONFIG levels of the CLI. You can configure the Ruckus device to prompt only for a password. The device uses the username entered at login, if one is available. If no username was entered at login, the device prompts for both username and password.

The following example configures the Ruckus device to prompt only for a password when a user attempts to gain Super User access to the Privileged EXEC and CONFIG levels of the CLI.

```
device# configure terminal
device(config)# aaa authentication enable implicit-user
```

RADIUS authorization

Ruckus devices support RADIUS authorization for controlling access to management functions in the CLI. Two kinds of RADIUS authorization are supported:

- Exec authorization determines a user privilege level when they are authenticated
- Command authorization consults a RADIUS server to get authorization for commands entered by the user

Configuring exec authorization

When RADIUS exec authorization is performed, the Ruckus device consults a RADIUS server to determine the privilege level of the authenticated user. To configure RADIUS exec authorization on the Ruckus device, enter the following command.

```
device(config)#aaa authorization exec default radius
```

If you specify **none** , or omit the **aaa authorization exec** command from the device configuration, no exec authorization is performed.

NOTE

If the **aaa authorization exec default radius** command exists in the configuration, following successful authentication the device assigns the user the privilege level specified by the foundry-privilege-level attribute received from the RADIUS server. If the **aaa authorization exec default radius** command does not exist in the configuration, then the value in the foundry-privilege-level attribute is ignored, and the user is granted Super User access.Also note that in order for the **aaa authorization exec default radius** command to work, either the **aaa authentication enable default radius** command, or the **aaa authentication login privilege-mode** command must also exist in the configuration.

Configuring command authorization

When RADIUS command authorization is enabled, the Ruckus device consults the list of commands supplied by the RADIUS server during authentication to determine whether a user can issue a command he or she has entered.

You enable RADIUS command authorization by specifying a privilege level for which commands require authorization.

To configure the Ruckus device to perform authorization for the commands available at the Super User privilege level (that is, all commands on the device), enter the commands shown in the following example.

```
device# configure terminal
device(config)# aaa authorization commands 0 default radius
```

The example sets the privilege level to 0, for Super User authorization, and configures RADIUS authorization.

The privilege-level parameter can be one of the following:

- **0** - Authorization is performed (that is, the Ruckus device looks at the command list) for commands available at the Super User level (all commands).
- **4** - Authorization is performed for commands available at the Port Configuration level (port-config and read-only commands).
- **5** - Authorization is performed for commands available at the Read Only level (read-only commands).

NOTE
RADIUS command authorization can be performed only for commands entered from Telnet or SSH sessions, or from the console. No authorization is performed for commands entered at the Web Management Interface.

NOTE
Since RADIUS command authorization relies on the command list supplied by the RADIUS server during authentication, you cannot perform RADIUS authorization without RADIUS authentication.

Command authorization and accounting for console commands

The Ruckus device supports command authorization and command accounting for CLI commands entered at the console.

CAUTION
If you have previously configured the device to perform command authorization using a RADIUS server, entering the enable aaa console command may prevent the execution of any subsequent commands entered on the console. This happens because RADIUS command authorization requires a list of allowable commands from the RADIUS server. This list is obtained during RADIUS authentication. For console sessions, RADIUS authentication is performed only if you have configured Enable authentication and specified RADIUS as the authentication method (for example, with the aaa authentication enable default radius command). If RADIUS authentication is never performed, the list of allowable commands is never obtained from the RADIUS server. Consequently, there would be no allowable commands on the console.

To configure the device to perform command authorization and command accounting for console commands, enter the following command.

```
device# configure terminal
device(config)# enable aaa console
```

RADIUS accounting

Ruckus devices support RADIUS accounting for recording information about user activity and system events. When you configure RADIUS accounting on a Ruckus device, information is sent to a RADIUS accounting server when specified events occur, such as when a user logs into the device or the system is rebooted.

Configuring RADIUS accounting for Telnet/SSH (Shell) access

To send an Accounting Start packet to the RADIUS accounting server when an authenticated user establishes a Telnet or SSH session on the Ruckus device and an Accounting Stop packet when the user logs out, enter the following command.

```
device# configure terminal
device(config)# aaa accounting exec default start-stop radius
```

Configuring RADIUS accounting for CLI commands

You can configure RADIUS accounting for CLI commands by specifying a privilege level.

To configure the Ruckus device to perform RADIUS accounting for the commands available at the Super User privilege level (that is; all commands on the device), enter the following command:

```
device# configure terminal
device(config)# aaa accounting commands 0 default start-stop radius
```

The example configures RADIUS accounting for commands at the Super User level (0).

An Accounting Start packet is sent to the RADIUS accounting server when a user enters a command, and an Accounting Stop packet is sent when the service provided by the command is completed.

The privilege-level parameter can be one of the following values:

- **0** - Records commands available at the Super User level (all commands)
- **4** - Records commands available at the Port Configuration level (port-config and read-only commands)
- **5** - Records commands available at the Read Only level (read-only commands)

NOTE
If authorization is enabled and the command requires authorization, authorization is performed before accounting takes place. If authorization fails for the command, no accounting takes place.

Configuring RADIUS accounting for system events

You can configure RADIUS accounting to record when system events occur on the Ruckus device. System events include reboots and changes to the active configuration.

To configure RADIUS accounting for system events, enter the following command.

```
device# configure terminal
device(config)# aaa accounting system default start-stop radius
```

The example configures the device to send an Accounting Start packet to the RADIUS accounting server when a system event occurs and an Accounting Stop packet when the system event has completed.

RADIUS accounting for 802.1X authentication and MAC authentication

802.1X accounting and MAC authentication accounting enable the recording of information about the clients that were successfully authenticated and allowed access to the network. When 802.1X authentication accounting or MAC authentication accounting is enabled, the device sends accounting information to the RADIUS server when a session begins, and a message when the session ends.

An Accounting Start packet is sent to the RADIUS server when a user is successfully authenticated. The Start packet indicates the start of a new session and contains the user MAC address and physical port number. The 802.1X or MAC session state will change to Authenticated and Permit after receiving a response from the accounting server for the accounting Start packet. If the Accounting service is not available, the session status will change to Authenticated and Permit after a RADIUS timeout. The device will retry authentication requests three times (the default), or the number of times configured on the device.

When 802.1X authentication accounting or MAC authentication accounting is enabled on the Ruckus device, it sends the following information to a RADIUS server whenever an authenticated client (user) logs into or out of the Ruckus device:

- The user name
- The session ID
- The user MAC address
- The authenticating physical port number
- Input bytes/octets
- Input packets
- Output octets/bytes
- Output packets
- VLAN-ID
- Elapsed Session-time
- Framed IP Address (client IP address)

An Accounting Stop packet is sent to the RADIUS server when one of the following events occur:

- The user logs off
- The port goes down
- The port is disabled
- The user fails to re-authenticate after a RADIUS timeout
- The 802.1X port control-auto configuration changes
- The MAC session clears (through use of the **clear dot1x mac-session** or **clear mac-auth sessions** commands)

The Accounting Stop packet indicates the end of the session and the time the user logged out.

Interim RADIUS accounting update for 802.1X authentication and MAC authentication

Apart from setting the device to send the Accounting Start and Accounting Stop messages, you can configure the device to send interim updates of accounting messages to the RADIUS server at regular intervals. The interim update message sends the status of an active session and current user statistics including the duration of the current session and information on current data usage at regular intervals.

The interim update message is useful for billing longer sessions. Normally, RADIUS server uses the information from both Accounting Start and Accounting Stop packets to generate accounting information for billing or other purpose. Information such

as session time, number of bytes transferred and so on that are critical for billing are available only in the Accounting Stop packets. If the device becomes unavailable due to any unexpected reason, the data required for billing the sessions initiated on the device will not be recorded. In such scenarios, interim update is useful which sends information regarding a specific session to the RADIUS server at regular intervals. For more information about the RADIUS attributes to support interim updates, refer to the list of supported RADIUS attributes in "Flexible Authentication".

The interim update and the interval between each interim update can also be configured on the device using the **radius-server accounting interim-updates** and **radius-server accounting interim-interval** commands respectively. The RADIUS accounting for 802.1X authentication and MAC authentication accepts either the interim update interval value configured using the RADIUS attribute or the interval time value set on the device, whichever is higher.

Enabling RADIUS accounting for 802.1X authentication and MAC authentication

To enable RADIUS accounting for 802.1X authentication and MAC accounting, perform the following steps.

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

```
device# configure terminal
```

2. Enable accounting for 802.1X authentication or MAC authentication by entering respective commands from the following steps.
   - Enable 802.1X authentication accounting.

```
device(config)# aaa accounting dot1x default start-stop radius
```

   - Enable MAC authentication accounting.

```
device(config)# aaa accounting mac-auth default start-stop radius
```

3. Enable interim updates and set accounting update intervals for 802.1X authentication accounting and MAC authentication accounting.

```
device(config)# radius-server accounting interim-updates
device(config)# radius-server accounting interim-interval 10
```

Dead RADIUS server detection

Ruckus ICX devices support authentication using up to eight RADIUS servers, including those used for authentication and for management.

The device tries to use the servers in the order they are added to the device configuration. If one RADIUS server times out (does not respond), the Ruckus ICX device tries the next one in the list. Servers are tried in the same sequence each time there is a request, and if multiple servers are unavailable or not responding, authentication delay results.

The RADIUS servers that are unavailable or that have stopped responding can be detected and marked as dead servers using the **radius-server test** command. To test the availability of the server, an Access-Request message is sent to the RADIUS server using a nonexistent username; that is, a username that is not configured on the server, so that the server responds with an Access-Reject message if the server is available. If the Ruckus ICX device does not receive a response from a RADIUS server within a specified time limit and number of retries, the RADIUS server is marked as dead. The time limit and number of retries can be manually configured using the **radius-server timeout** and **radius-server retransmit** commands respectively. If the parameters are not manually configured, the Ruckus ICX device applies the default value of 3 seconds with a maximum of 3 retries. The

interval at which the test message is sent to check the status of the server can be configured using the **radius-server dead-time** command.

The following example demonstrates configuring and confirming dead server detection.

```
device# configure terminal
device(config)# radius-server test sample
device(config)# radius-server dead-time  5
device(config)# exit
device# show radius server
-------------------------------------------------------------------------------
Server                      Type      Opens     Closes   Timeouts   Status
-------------------------------------------------------------------------------
10.20.226.113                any        471        247          1   active
10.20.226.114                any        471        247          1   dead
```

When dead RADIUS server detection is configured, all configured RADIUS servers are monitored on a regular basis from system startup. When a RADIUS server times out (does not respond), it is marked as a dead server. When the status of a RADIUS server tagged for 802.1x or MAC authentication changes, the new status is broadcast to all units in the stack because 802.1x and MAC authentication is performed locally on all units.

Source address configuration for RADIUS packets

You can designate the lowest-numbered IP address configured on Ethernet port, loopback interface, or virtual interface as the source IP address for all RADIUS packets from the Layer 3 Switch.

When a source interface is configured, management applications use the lowest configured IP address of the specified interface as the source IP address in all the outgoing packets. If the configured interface is not part of the management VRF, the response packet does not reach the destination.

The RADIUS source interface configuration command **ip radius source-interface** should be compatible with the management VRF configuration.

Any change in the management VRF configuration takes effect immediately for the RADIUS client.

The following example configures an Ethernet interface as the source IP address for the RADIUS client.

```
device(config)# ip radius source-interface ethernet 1/1/1
```

The following example configures a loopback interface as the source IP address for the RADIUS client.

```
device(config)# ip radius source-interface loopback 1
```

Refer to the *Ruckus FastIron Command Reference* for details on the **ip radius source-interface** command.

RADIUS dynamic authorizations

When a user or device is authenticated on the RADIUS server, the session can only be ended if the user or device logs out. There is no way to change the previously downloaded policies or configuration.

RFC 5176 addresses this issue by adding two more packet types to the current RADIUS standard: Disconnect Message and Change of Authorization. The Dynamic Authorization Client (DAC) server makes the requests to either delete the previously established sessions or replace the previous configuration or policies. Currently, these new extensions can be used to dynamically terminate or authorize sessions that are authenticated through MAC authentication, 802.1x authentication, or Web authentication.

RADIUS Disconnect Message and CoA events

Describes the events that take place during Disconnect Message and Change of Authorization.

The following events occur when a disconnect message is sent out by the Dynamic Authorization Client (DAC):

- A disconnect request packet is sent by the Dynamic Authorization Client (DAC) to terminate the session on the NAS (Network Access Server) and discard the associated session contexts.
- The request identifies the NAS and the session to be removed. This packet is sent to UDP port 3799 on the NAS.
- The NAS responds with a disconnect-ACK, if the session is identified, removed, and no longer valid.
- The NAS sends a disconnect-NAK if it is unable to disconnect the session.

The following events occur when a change of authorization request packet is sent by the Dynamic Authorization Client (DAC):

- A change of authorization request packet is sent by the Dynamic Authorization Client (DAC) to change the session authorizations on the NAS. This is used to change the filters, such as Layer 3 ACLs.
- The request identifies the NAS and the sessions to be authorized. The request carries the filter ID attribute (type 11). The attribute will specify the IP ACL that is to be applied. This packet is sent to UDP port 3799 on the NAS.
- The NAS responds with a CoA-ACK (CoA acknowledgment) if the session is identified and authorized with new filters. It sends a CoA non-acknowledgment, if it is unable to apply the filters on the session.

  NOTE
  Currently, Ruckus ICX devices support applying ACLs to those sessions that have IP ACLs applied in the previous Authorization. You cannot use CoA to configure IP ACLs on a session that is not authenticated with an ACL.

Enabling RADIUS CoA and Disconnect Message handling

To enable RADIUS Disconnect Message and CoA handling, complete the following steps:

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **aaa authorization coa enable** command.

   ```
   device(config)# aaa authorization coa enable
   ```

3. Configure the shared secret key required between the CoA client and the device as shown in the following example. Enter the **radius-client coa host** command followed by the CoA host address, the word **key**, and the string that is required between the CoA client and the ICX device.

   ```
   device(config)# radius-client coa host 10.24.65.6 key fastiron123
   ```

   The example configures the shared secret key fastiron123 to be used between the ICX device and the CoA host with the IP address 10.24.65.6.

   NOTE
   An IPv6 address can be used when preceded by the keyword **ipv6**.

Supported IETF attributes in RFC 5176

Describes the supported IETF attributes and error clause values.

Some of the supported IETF attributes are listed in the following table.

TABLE 9 Supported IETF attributes

Attribute Name	Attribute Number	Description
NAS-IP-Address	4	IPv4 address of NAS
NAS-Identifier	32	The port, where the session is terminated
NAS-IPv6-Address	95	IPv6 address of NAS
Calling-Station-Id	31	Link address from which sessions are connected
Filter-ID	11	Indicates the name of a data filter list to be applied for the sessions that the identification attributes map to.

Error clause values

When the NAS cannot honor the disconnect message and CoA requests, the NAS sends corresponding NAK responses. These responses must include the error clause attribute to provide more details on the possible cause of the problem. The format of this error clause attribute is the same as any other attribute and the value field consists of a 4-byte integer.

The error cause attribute values are organized in the following series:

- 0-199 Reserved
- 200-299 Successful completion
- 300-399 Reserved
- 400-499 Fatal errors committed by Dynamic Authorization Client (DAC)
- 500-599 Fatal errors committed by Dynamic Authorization Server (DAS)

TABLE 10 Error clause values

Value	Description
401	Unsupported attribute
402	Missing attribute
403	NAS identification mismatch
404	Invalid Request
405	Unsupported services
407	Invalid attribute value
501	Administratively prohibited (used when a CoA request or disconnect message is ignored because of configuration)
503	Session context not found
506	Resources unavailable

Displaying RADIUS configuration information

The **show aaa** command displays information about all TACACS/TACACS+ and RADIUS servers identified on the device.

```
device#show aaa
Tacacs+ key: foundry
```

```
Tacacs+ retries: 1
Tacacs+ timeout: 15 seconds
Tacacs+ Server: 10.95.6.90 Port:49:
                opens=6 closes=3 timeouts=3 errors=0
                packets in=4 packets out=4
no connection
Radius key: networks
Radius retries: 3
Radius timeout: 3 seconds
Radius Server: 10.95.6.90 Auth Port=1812 Acct Port=1813:
                opens=2 closes=1 timeouts=1 errors=0
                packets in=1 packets out=4
no connection
```

The following table describes the RADIUS information displayed by the **show aaa** command.

TABLE 11 Output of the show aaa command for RADIUS

Field	Description
Radius default key	The setting configured with the **radius-server key** command. At the Super User privilege level, the actual text of the key is displayed. At the other privilege levels, a string of periods (....) is displayed instead of the text.
Radius retries	The setting configured with the **radius-server retransmit** command.
Radius timeout	The setting configured with the **radius-server timeout** command.
Radius Server	For each RADIUS server, the IP address, and the following statistics are displayed: Auth Port RADIUS authentication port number (default 1812) Acct Port RADIUS accounting port number (default 1813) • opens - Number of times the port was opened for communication with the server • closes - Number of times the port was closed normally • timeouts - Number of times port was closed due to a timeout • errors - Number of times an error occurred while opening the port • packets in - Number of packets received from the server • packets out - Number of packets sent to the server
connection	The current connection status. This can be "no connection" or "connection active".

The **show web connection** command displays the privilege level of Web Management Interface users.

Security Vulnerability

SSL security

The Ruckus ICX device supports Transport Level Security. By default, all TLS versions will be supported on devices that act as an HTTP server.

When enabled, the SSL protocol uses digital certificates and public-private key pairs to establish a secure connection to the Ruckus ICX device. Digital certificates serve to prove the identity of a connecting client, and public-private key pairs provide a means to encrypt data sent between the device and the client.

Configuring SSL consists of the following tasks:

1. Optionally enabling the SSL server on the Ruckus ICX device

   NOTE
   The SSL server is automatically enabled when an SSL certificate is generated.

2. Importing an RSA certificate and private key file from a client (optional)
3. Generating a certificate

Enabling the SSL server on the device

To enable the SSL server on the Ruckus ICX device, enter the following command.

```
device# configure terminal
device(config)# web-management https
```

The example enables the SSL server for HTTPS. Use the **http** parameter in the same command to enable HTTP servers.

To disable both HTTP and HTTPs servers, enter the following command.

```
device# configure terminal
device(config)# no web-management
```

Specifying a port for SSL communication

By default, SSL protocol exchanges occur on TCP port 443. You have the option to change the port number used for SSL communication.

The following example configures the device to use TCP port 334 for SSL communication.

```
device# configure terminal
device(config)# ip ssl port 334
```

Changing the SSL server certificate key size

The default key size for Ruckus-issued and imported digital certificates is 1024 bits. You can change the default key size to a value of 2048 or 4096 bits. For example, to change the key size to 4096 bits, enter the following command.

```
device# configure terminal
device(config)# ip ssl cert-key-size 4096
```

NOTE
The SSL server certificate key size applies only to digital certificates issued by Ruckus and does not apply to imported certificates.

Support for SSL digital certificates larger than 2048 bits

Ruckus ICX devices have the ability to store and retrieve SSL digital certificates that are up to 4000 bits in size.

Support for SSL certificates larger than 2048 bits is automatically enabled. You do not need to perform any configuration procedures to enable it.

Importing digital certificates and RSA private key files

To allow a client to communicate with another Ruckus ICX device using an SSL connection, you must configure a set of digital certificates and RSA public-private key pairs on the device. A digital certificate is used to identify the connecting client to the server. The certificate contains information about the issuing Certificate Authority as well as a public key. You can import digital certificates and private keys from a server, or you can allow the Ruckus ICX device to create them.

If you want to allow the Ruckus ICX device to create the digital certificates, refer to Generating an SSL certificate on page 76. If you choose to import an RSA certificate and private key file from a client, you can use TFTP to transfer the files.
To import a digital certificate using TFTP, enter the **ip ssl certificate-data-file tftp** command followed by the IP address of the TFTP server and the filename of the certificate as shown in the following example.

```
device# configure terminal
device(config)# ip ssl certificate-data-file tftp 192.168.9.210 certfile
```

The example imports the certificate file named certfile from the TFTP server with the IP address 192.168.9.210.

To import an RSA private key from a client using TFTP, enter the **ip ssl private-key-file tftp** command followed by the IP address of the TFTP server and the name of the key file as shown in the following example.

```
device# configure terminal
device(config)# ip ssl private-key-file tftp 192.168.9.210 keyfile
```

The example imports the private key file named keyfile from the TFTP server with the IP address 192.168.9.210.

NOTE
The RSA key can contain a maximum of 4096 bits.

Generating an SSL certificate

After you have imported the digital certificate, it should automatically generate.
If the certificate does not automatically generate, enter the following command to generate it.

```
device(config)#crypto-ssl certificate generate
```

Deleting the SSL certificate

To delete the SSL certificate, enter the following command.

```
device(config)# crypto-ssl certificate zeroize
```

TLS support

By default, all TLS versions such as TLS 1.0, TLS 1.1, and TLS 1.2 are supported on devices that act as an HTTP server.

For devices that act as the SSL client or the syslog, OpenFlow, RADIUS, or secure AAA client, the TLS version is decided based on the server support.

You can configure the minimum TLS version on FastIron devices using the **ip ssl min-version** command. The TLS version configured as the minimum version and all the later versions are supported to establish the connection. For example, if TLS 1.1 version is configured as the minimum version, both TLS 1.1 and TLS 1.2 versions are supported. For devices that act as an SSL server or HTTPS server, the default connection is with TLS1.2.

You can use the **show ip ssl** command to identify the TLS version that is configured on the device.

For TLS support of RADIUS, the RADIUS server checks the certificate to make sure that the user connecting for authentication is not being intercepted. The RADIUS server then determines that the server and client are using the same encryption types. Then the RADIUS server and device send each other unique codes to use when encrypting the data traffic.

Authentication-method lists

To implement one or more authentication methods for securing access to the device, you configure authentication-method lists that set the order in which the authentication methods are consulted.

In an authentication-method list, you specify the access method (Telnet, SNMP, and so on) and the order in which the device tries one or more of the following authentication methods:

- Local Telnet login password
- Local password for the Super User privilege level
- Local user accounts configured on the device
- Database on a TACACS or TACACS+ server
- Database on a RADIUS server
- No authentication

NOTE
The TACACS/TACACS+, RADIUS, and Telnet login password authentication methods are not supported for SNMP access.

NOTE
To authenticate Telnet access to the CLI, you also must enable the authentication by entering the **enable telnet authentication** command at the global CONFIG level of the CLI. You cannot enable Telnet authentication using the Web Management Interface.

NOTE
You do not need an authentication-method list to secure access based on ACLs or a list of IP addresses. Refer to ACL usage to restrict remote access on page 29 or Remote access restrictions on page 31.

In an authentication-method list for a particular access method, you can specify up to seven authentication methods. If the first authentication method is successful, the software grants access and stops the authentication process. If the access is rejected by the first authentication method, the software denies access and stops checking.

However, if an error occurs with an authentication method, the software tries the next method on the list, and so on. For example, if the first authentication method is the RADIUS server, but the link to the server is down, the software will try the next authentication method in the list.

NOTE
If an authentication method is working properly and the password (and user name, if applicable) is not known to that method, this is not an error. The authentication attempt stops, and the user is denied access.

The software will continue this process until either the authentication method is passed or the software reaches the end of the method list. If the Super User level password is not rejected after all the access methods in the list have been tried, access is granted.

Configuration considerations for authentication-method lists

- For CLI access, you must configure authentication-method lists if you want the device to authenticate access using local user accounts or a RADIUS server. Otherwise, the device will authenticate using only the locally based password for the Super User privilege level.
- If the last local user is removed from the device, the following configuration changes occur:
  - AAA configuration specific to local authentication is removed.
  - When no other AAA mechanism (RADIUS or TACACS) is configured, "enable aaa console" configuration is removed from the device.
  - The method "local" becomes unavailable in the AAA authentication-method list configuration.
- When no authentication-method list is configured specifically for Web management access, the device performs authentication using the SNMP community strings:
  - For read-only access, you can use the user name “get” and the password “public”.
  - There is no default read-write community string. Thus, by default, you cannot open a read-write management session using the Web Management Interface. You first must configure a read-write community string using the CLI. Then you can log on using “set” as the user name and the read-write community string you configure as the password. Refer to TACACS and TACACS+ security on page 41.
- If you configure an authentication-method list for Web management access and specify “local” as the primary authentication method, users who attempt to access the device using the Web Management Interface must supply a user name and password configured in one of the local user accounts on the device. The user cannot access the device by entering “set” or “get” and the corresponding SNMP community string.

Examples of authentication-method lists

The following examples show how to configure authentication-method lists. In these examples, the primary authentication method for each is local. The device authenticates access attempts using the locally configured usernames and passwords.

Example 1

To configure an authentication-method list for the Web Management Interface, enter the **aaa authentication web-server default** command followed by the primary authentication method (and, as an option, authentication methods you want to designate as alternates).

```
device# configure terminal
device(config)# aaa authentication web-server default local
```

The example configures the device to use local user accounts to authenticate access to the device through the Web Management Interface. No alternate access methods are configured. If the device does not have a user account that matches the user name and password entered, the user is not granted access.

Example 2

To configure an authentication-method list for SNMP, enter the **aaa authentication snmp-server default** command followed by the primary authentication method (and, as an option, authentication methods you want to designate as alternates).

```
device# configure terminal
device(config)# aaa authentication snmp-server default local
```

The example allows certain incoming SNMP SET operations to be authenticated using the locally configured usernames and passwords. No alternate access methods are configured. When the **aaa authentication snmp-server default** command is enabled, community string validation is not performed for incoming SNMP V1 and V2c packets. This command takes effect as long as the first varbind for SNMP packets is set to one of the following:

- snAgGblPassword=" username password " (for AAA method local)
- snAgGblPassword=" password " (for AAA method line, enable)

NOTE
Certain SNMP objects require additional validation. These objects include but are not limited to: **snAgReload**, **snAgWriteNVRAM**, **snAgConfigFromNVRAM**, **snAgImgLoad**, **snAgCfgLoad** and **snAgGblTelnetPassword**. For more information, refer to **snAgGblPassword** in the *Ruckus FastIron MIB Reference*.

If AAA is set up to check both the username and password, the string contains the username, followed by a space and then the password. If AAA is set up to authenticate with the current Enable or Line password, the string contains the password only.

NOTE
The previous configuration can be overridden by the command **no snmp-server pw-check** , which disables password checking for SNMP SET requests.

Example 3

To configure an authentication-method list for the Privileged EXEC and CONFIG levels of the CLI, enter the **aaa authentication enable default** command followed by the primary authentication method (and, as an option, authentication methods you want to designate as alternates).

```
device# configure terminal
device(config)# aaa authentication enable default local
```

The example configures the device to use local user accounts to authenticate attempts to access the Privileged EXEC and CONFIG levels of the CLI. No alternate methods of authentication are configured.

Example 4

To configure the device to use a RADIUS server first to authenticate access to the Privileged EXEC and CONFIG levels of the CLI, enter the **aaa authentication enable default** command followed by the keyword radius (and, as an option, authentication methods you want to designate as alternates).

```
device# configure terminal
device(config)# aaa authentication enable default radius local
```

The example configures the device to consult a RADIUS server first to authenticate attempts to access the Privileged EXEC and CONFIG levels of the CLI and then to consult the local user accounts if the RADIUS server is unavailable.

Authentication methods available

When you enter an authentication list for any of the AAA authentication commands, the first authentication listed is the primary method of authentication. Additional methods of authentication listed are used as alternates if the primary authentication fails.

The following table lists available values that can be entered as authentication methods.

NOTE
TACACS/TACACS+ and RADIUS authentication options are supported only with the **enable** and **login** access parameters.

TABLE 12 Authentication method values

Method parameter	Description
line	Authenticate using the password you configured for Telnet access. The Telnet password is configured using the **enable telnet password...** command. Refer to Setting a Telnet password on page 19.
enable	Authenticate using the password you configured for the Super User privilege level. This password is configured using the **enable super-user-password...** command. Refer to Setting passwords for management privilege levels on page 19.
local	Authenticate using a local user name and password you configured on the device. Local user names and passwords are configured using the **username...** command. Refer to Local user account configuration on page 26.
tacacs	Authenticate using the database on a TACACS server. You also must identify the server to the device using the **tacacs-server** command.
tacacs+	Authenticate using the database on a TACACS+ server. You also must identify the server to the device using the **tacacs-server** command.
radius	Authenticate using the database on a RADIUS server. You also must identify the server to the device using the **radius-server** command. Refer to RADIUS security on page 55.
none	Do not use any authentication method. The device automatically permits access.

Secure Shell (SSH)

SSH version 2 overview

Secure Shell (SSH) is a mechanism for allowing secure remote access to management functions on a Ruckus device. SSH provides a function similar to Telnet. Users can log into and configure the device using a publicly or commercially available SSH client program, just as they can with Telnet. However, unlike Telnet, which provides no security, SSH provides a secure, encrypted connection to the device.

The Ruckus SSH2 implementation is compatible with all versions of the SSH2 protocol (2.1, 2.2, and so on). At the beginning of an SSH session, the Ruckus ICX device negotiates the version of SSH2 to be used. The highest version of SSH2 supported by both the Ruckus ICX device and the client is the version that is used for the session. Once the SSH2 version is negotiated, the encryption algorithm with the highest security ranking is selected to be used for the session.

Ruckus ICX devices also support Secure Copy (SCP) for securely transferring files between a Ruckus ICX device and SCP-enabled remote hosts.

NOTE
The SSH feature includes software that is copyright Allegro Software Development Corporation.

SSH2 is supported in the Layer 2 and Layer 3 codes.

SSH2 is a substantial revision of Secure Shell, comprising the following hybrid protocols and definitions:

- SSH Transport Layer Protocol
- SSH Authentication Protocol
- SSH Connection Protocol
- SECSH Public Key File Format
- SSH Fingerprint Format
- SSH Protocol Assigned Numbers
- SSH Transport Layer Encryption Modes
- SCP/SSH URI Format

Tested SSH2 clients

The following SSH clients have been tested with SSH2:

- SSH Secure Shell 3.2.3
- Van Dyke SecureCRT 5.2.2
- F-Secure SSH Client 5.3 and 6.0

- PuTTY 0.62

  NOTE
  SSH session may drop when using PuTTY on Windows system and left idle for more than 45 minutes.

- OpenSSH 4.3p2
- Ruckus ICX SSH Client

  NOTE
  Supported SSH client public key sizes are 1024 or 2048 bits for DSA keys and RSA keys.

SSH2 supported features

SSH2 (Secure Shell version 2 protocol) provides an SSH server and an SSH client. The SSH server allows secure remote access management functions on a Ruckus device. SSH provides a function that is similar to Telnet, but unlike Telnet, SSH provides a secure, encrypted connection.

Ruckus SSH2 support includes the following:

- Key exchange methods are **diffie-hellman-group14-sha1** and **diffie-hellman-group1-sha1** .
- The supported public key algorithms are **ssh-dss and ssh-rsa** .
- Encryption is provided with 3des-cbc , aes128-cbc , aes192-cbc, aes256-cbc, aes128-ctr, aes192-ctr, aes256-ctr. AES encryption has been adopted by the U.S. Government as an encryption standard.
- Data integrity is ensured with **hmac-sha1.**
- Supported authentication methods are **Password** , **interactive,** and **Key authentication.**
- Five inbound SSH connection at one time are supported.
- Five outbound SSH is supported.

SSH2 unsupported features

The following are not supported with SSH2:

- Compression
- TCP/IP port forwarding, X11 forwarding, and secure file transfer
- SSH version 1

SSH2 authentication types

The Ruckus implementation of SSH2 supports the following types of user authentication:

- DSA challenge-response authentication , where a collection of public keys are stored on the device. Only clients with a private key that corresponds to one of the stored public keys can gain access to the device using SSH.
- RSA challenge-response authentication , where a collection of public keys are stored on the device. Only clients with a private key that corresponds to one of the stored public keys can gain access to the device using SSH.
- Password authentication , where users attempting to gain access to the device using an SSH client are authenticated with passwords stored on the device or on a TACACS or TACACS+ server or a RADIUS server.
- Interactive-authentication
- Keyboard-interactive authentication

Configuring SSH2

You can configure the device to use any combination of these authentication types. The SSH server and client negotiate which type to use.

To configure SSH2, follow these steps:

1. Generate a host Digital Signature Algorithm (DSA) or Ron Rivest, Adi Shamir and Leonard Adleman Algorithm (RSA), and private key pair for the device.

   See the section Enabling and disabling SSH by generating and deleting host keys on page 83.
2. Configure DSA or RSA challenge-response authentication.

   See the section Configuring DSA or RSA challenge-response authentication on page 85.
3. Set optional parameters.

   See the section Optional SSH parameters on page 87.

Enabling and disabling SSH by generating and deleting host keys

To enable SSH, you generate a DSA or RSA host key on the device. The SSH server on the Ruckus ICX device uses this host DSA or RSA key, along with a dynamically generated server DSA or RSA key pair, to negotiate a session key and encryption method with the client trying to connect to it.

While the SSH listener exists at all times, sessions can not be started from clients until a host key is generated. After a host key is generated, clients can start sessions.

To disable SSH, you delete all of the host keys from the device.

When a host key is generated, it is saved to the flash memory of all management modules. When a host key is is deleted, it is deleted from the flash memory of all management modules.

The time to initially generate SSH keys varies depending on the configuration, and can be from a under a minute to several minutes.

SSHv2 RSA host key format is different between FastIron 07.x.xx, 08.0.00 and 08.0.00a software versions .

- When you upgrade from FastIron 7.x.xx, 8.0.00 to 8.0.00a software version , if RSA key is present in FastIron 7.x.xx or 8.0.00 software version, same size will be regenerated in FastIron 08.0.00a software version. Old SSHv2 host key is retained unless they are cleared by the **crypto key zeroize** command.
- When you downgrade the FastIron software from version 8.0.00a to 8.0.00 or 07.x.xx, consider the following scenarios:
  - SSHv2 RSA host key created in FastIron 7.x.xx or 8.0.00 software version and retained in FastIron 8.0.00a-- In this case, booting up with FastIron 7.x.xx or 8.0.00 software versions reads the old format SSHv2 RSA host keys and enables the SSHv2 RSA server on the switch.
  - SSHv2 RSA host key created in FastIron 8.0.00a--In this case, booting up with FastIron 7.x.xx or 8.0.00 software versions does not read the new format SSHv2 RSA host keys and SSHv2 server is not enabled on the switch.

SSH host keys created with DSA method is interoperable between FastIron 7.x.xx, 8.0.00 and 8.0.00a software versions.

Generating and deleting a DSA key pair

To generate a DSA key pair, enter the following command.

```
device(config)# crypto key generate dsa
```

The **generate** keyword places a host key pair in the flash memory and enables SSH on the device, if it is not already enabled. The **dsa** keyword specifies a DSA host key pair. This keyword is optional. If you do not enter it, the command **crypto key generate** generates a DSA key pair by default.

To delete the DSA host key pair, enter the following command.

```
device(config)# crypto key zeroize dsa
```

The **zeroize** keyword deletes the host key pair from the flash memory. This disables SSH if no other server host keys exist on the device.

For more information on the **crypto key zeroize** command, refer to Deleting DSA and RSA key pairs on page 84.

Generating and deleting an RSA key pair

To generate an RSA key pair, enter the **crypto key generate rsa** command as shown in the following example.

```
device# configure terminal
device(config)# crypto key generate rsa
```

The **generate** keyword places an RSA host key pair in the flash memory and enables SSH on the device, if it is not already enabled. The example generates an RSA key with the default modulus of 1024 bits. To generate a modulus 2048 key, enter the following command.

```
device# configure terminal
device(config)# crypto key generate rsa modulus 2048
```

To delete the RSA host key pair, enter the following command.

```
device(config)# crypto key zeroize rsa
```

The **zeroize** keyword deletes the RSA host key pair from the flash memory. This disables SSH if no other authentication keys exist on the device.

Deleting DSA and RSA key pairs

To delete DSA and RSA key pairs from the flash memory, enter the following command:

```
device(config)# crypto key zeroize
```

The **zeroize** keyword deletes the host key pair from the flash memory. This disables SSH.

Providing the public key to clients

The host DSA or RSA key pair is stored in the system-config file of the Ruckus ICX device. Only the public key is readable. Some SSH client programs add the public key to the known hosts file automatically. In other cases, you must manually create a known hosts file and place the public key of the Ruckus ICX device in it.

If you are using SSH to connect to a Ruckus ICX device from a UNIX system, you may need to add the public key on the Ruckus ICX device to a “known hosts” file on the client UNIX system; for example, $HOME/.ssh/known_hosts. The following is an example of an entry in a known hosts file.

```
AAAAB3NzaC1kc3MAAACBAPY8ZOHY2yFSJA6XYC9HRwNHxaehvx5wOJ0rzZdzoSOXxbET
W6ToHv8D1UJ/
z+zHo9Fiko5XybZnDIaBDHtblQ+Yp7StxyltHnXF1YLfKD1G4T6JYrdH YI14Om
1eg9e4NnCRleaqoZPF3UGfZia6bXrGTQf3gJq2e7Yisk/gF+1VAAAAFQDb8D5cv
wHWTZDPfX0D2s9Rd7NBvQAAAIEAlN92+Bb7D4KLYk3IwRbXblwXdkPggA4pfdtW9v
GfJ0/RHd+NjB4eo1D+0dix6tXwYGN7PKS5R/FXPNwxHPapcj9uL1Jn2AWQ2dsknf+i/FAA
vioUPkmdMc0zuWoSOEsSNhVDtX3WdvVcGcBq9cetzrtOKWOocJmJ80qadxTRHtUAAACB
```

```
AN7CY+KKv1gHpRzFwdQm7HK9bb1LAo2KwaoXnadFgeptNBQeSXG1vO+JsvphVMBJc9HS
n24VYtYtsMu74qXviYjziVucWKjjKEb11juqnF0GDlB3VVmxHLmxnAz643WK42Z7dLM5
sY29ouezv4Xz2PuMch5VGPP+CDqzCM4loWgV
```

Configuring DSA or RSA challenge-response authentication

With DSA or RSA challenge-response authentication, a collection of clients' public keys are stored on the Ruckus device. Clients are authenticated using these stored public keys. Only clients that have a private key that corresponds to one of the stored public keys can gain access to the device using SSH.

When DSA or RSA challenge-response authentication is enabled, the following events occur when a client attempts to gain access to the device using SSH:

1. The client sends its public key to the Ruckus device.
2. The Ruckus ICX device compares the client public key to those stored in memory.
3. If there is a match, the Ruckus device uses the public key to encrypt a random sequence of bytes.
4. The Ruckus device sends these encrypted bytes to the client.
5. The client uses its private key to decrypt the bytes.
6. The client sends the decrypted bytes back to the Ruckus device.
7. The Ruckus device compares the decrypted bytes to the original bytes it sent to the client. If the two sets of bytes match, it means that the client private key corresponds to an authorized public key, and the client is authenticated.

   Setting up DSA or RSA challenge-response authentication consists of the following steps.

Importing authorized public keys into the device

SSH clients that support DSA or RSA authentication normally provide a utility to generate a DSA or RSA key pair. The private key is usually stored in a password-protected file on the local host; the public key is stored in another file and is not protected. You must import the client public key for each client into the Ruckus ICX device.

Collect one public key of each key type (DSA and/or RSA) from each client to be granted access to the Ruckus device and place all of these keys into one file. This public key file may contain up to 16 keys. The following is an example of a public key file containing one public key:

```
---- BEGIN SSH2 PUBLIC KEY ----
Comment: DSA Public Key  AAAAB3NzaC1kc3MAAACBAPY8ZOHY2yFSJA6XYC9HRwNHxaehvx5wOJ0rzZdzoSOXxbET W6ToHv8D1UJ/
z+zHo9Fiko5XybZnDIaBDHtblQ+Yp7StxyltHnXF1YLfKD1G4T6JYrdH YI14Om
1eg9e4NnCRleaqoZPF3UGfZia6bXrGTQf3gJq2e7Yisk/gF+1VAAAAFQDb8D5cv
wHWTZDPfX0D2s9Rd7NBvQAAAIEAlN92+Bb7D4KLYk3IwRbXblwXdkPggA4pfdtW9v
GfJ0/RHd+NjB4eo1D+0dix6tXwYGN7PKS5R/FXPNwxHPapcj9uL1Jn2AWQ2dsknf+i/FAA
vioUPkmdMc0zuWoSOEsSNhVDtX3WdvVcGcBq9cetzrtOKWOocJmJ80qadxTRHtUAAACB
AN7CY+KKv1gHpRzFwdQm7HK9bb1LAo2KwaoXnadFgeptNBQeSXG1vO+JsvphVMBJc9HS
n24VYtYtsMu74qXviYjziVucWKjjKEb11juqnF0GDlB3VVmxHLmxnAz643WK42Z7dLM5
sY29ouezv4Xz2PuMch5VGPP+CDqzCM4loWgV
---- END SSH2 PUBLIC KEY ----
```

NOTE
Each key in the public key file must begin and end with the first and last lines shown in the example. If your client does not include these lines in the public key, you must manually add them.

Import the authorized public keys into the Ruckus ICX device active configuration by loading this public key file from a TFTP server.

To load a public key file called pkeys.txt from a TFTP server, enter the **ip ssh pub-key-file** command followed by the keyword **tftp**, the IP address of the TFTP server, and the name of the public key file as shown in the following example.

```
device# configure terminal
device(config)# ip ssh pub-key-file tftp 10.168.1.234 pkeys.txt
```

To display the currently loaded public keys, enter the following command.

```
device# show ip client-pub-key
---- BEGIN SSH2 PUBLIC KEY ----
Comment: DSA Public Key  AAAAB3NzaC1kc3MAAACBAPY8ZOHY2yFSJA6XYC9HRwNHxaehvx5wOJ0rzZdzoSOXxbET W6ToHv8D1UJ/
z+zHo9Fiko5XybZnDIaBDHtblQ+Yp7StxyltHnXF1YLfKD1G4T6JYrdH YI14Om
1eg9e4NnCRleaqoZPF3UGfZia6bXrGTQf3gJq2e7Yisk/gF+1VAAAAFQDb8D5cv
wHWTZDPfX0D2s9Rd7NBvQAAAIEAlN92+Bb7D4KLYk3IwRbXblwXdkPggA4pfdtW9v
GfJ0/RHd+NjB4eo1D+0dix6tXwYGN7PKS5R/FXPNwxHPapcj9uL1Jn2AWQ2dsknf+i/FAA
vioUPkmdMc0zuWoSOEsSNhVDtX3WdvVcGcBq9cetzrtOKWOocJmJ80qadxTRHtUAAACB
AN7CY+KKv1gHpRzFwdQm7HK9bb1LAo2KwaoXnadFgeptNBQeSXG1vO+JsvphVMBJc9HS
n24VYtYtsMu74qXviYjziVucWKjjKEb11juqnF0GDlB3VVmxHLmxnAz643WK42Z7dLM5
sY29ouezv4Xz2PuMch5VGPP+CDqzCM4loWgV
---- END SSH2 PUBLIC KEY ----
```

To delete the public keys from the FastIron device, enter the **ip ssh pub-key-file remove** command and the name of the public key file as shown in the following example.

```
device# configure terminal
device(config)# ip ssh pub-key-file remove pkeys.txt
```

To clear the public keys from the buffers, enter the following command.

```
device# clear public-key
```

Enabling DSA or RSA challenge-response and password authentication

With DSA or RSA challenge-response authentication, a collection of clients' public keys are stored on the device. Clients are authenticated using these stored public keys. Only clients that have a private key that corresponds to one of the stored public keys can gain access to the device using SSH.

After the SSH server on the device negotiates a session key and encryption method with the connecting client, user authentication takes place. The implementation of SSH supports DSA or RSA challenge-response authentication and password authentication. You can deactivate one or both user authentication methods for SSH.

NOTE
Deactivating both authentication methods disables the SSH server entirely.

With password authentication, users are prompted for a password when they attempt to log in to the device (provided empty password logins are not allowed). If there is no user account that matches the username and password supplied by the user, the user is not granted access.

DSA and RSA challenge-response authentication is enabled by default. You can disable or re-enable it manually. To disable DSA and RSA challenge-response authentication, enter the following command.

To disable public key authentication, enter the following command.

```
device# configure terminal
device(config)# ip ssh key-authentication no
```

To re-enable public key authentication, enter the following command.

```
device# configure terminal
device(config)# ip ssh key-authentication yes
```

To enable keyboard-interactive authentication, enter the following command.

```
device# configure terminal
device(config)# ip ssh interactive-authentication yes
```

To disable keyboard interactive authentication, enter the following command.

```
device# configure terminal
device(config)# ip ssh interactive-authentication no
```

To enable DSA and RSA password authentication, enter the following command.

```
device# configure terminal
device(config)# ip ssh password-authentication yes
```

To disable DSA and RSA password authentication, enter the following command.

```
device# configure terminal
device(config)# ip ssh password-authentication no
```

Optional SSH parameters

You can adjust the following SSH settings on the Ruckus device:

- The number of SSH authentication retries
- The user authentication method the Ruckus device uses for SSH connections
- Key exchange method
- Whether the Ruckus device allows users to log in without supplying a password
- The port number for SSH connections
- The SSH login timeout value
- A specific interface to be used as the source for all SSH traffic from the device
- The maximum idle time for SSH sessions
- SSH rekey

Setting the number of SSH authentication retries

By default, the Ruckus device attempts to negotiate a connection with the connecting host three times. The number of authentication retries can be changed to between 1 - 5.

> **NOTE**
> The **ip ssh authentication-retries** command is not applicable on Ruckus ICX devices which acts as an SSH client. When the Ruckus ICX device acts as an SSH client and when you try to establish an SSH connection with wrong credentials, the session is not be established. The connection is terminated. The device does not check the SSH authentication retry configuration set using the **ip ssh authentication-retries** command. The command is applicable only to SSH clients like PUTTY, Secure CRT, and so on.

For example, the following command changes the number of authentication retries to 5.

```
device(config)# ip ssh authentication-retries 5
```

Supported key-exchange methods

Ruckus ICX SSH2 offers two key exchange methods such as diffie-hellman-group14-sha1 and diffie-hellman-group1-sha1 to establish an SSH connection. The Diffie-Hellman key exchange protocol parameters that the client and server choose is based on the client and server configuration. By default, when SSH clients connects, the ICX device as SSH server offers diffie-hellman-group14-sha1and diffie-hellman-group1-sha1 as the default key exchange methods and diffie-hellman-group14-sha1 will be given first priority. The diffie-hellman-group14-sha1 algorithm provides enhanced encryption of shared secrets between two devices and safeguards critical security parameters. As the diffie-hellman-group1-sha1 key exchange method is a weak algorithm, it can be disabled using the **no ip ssh key-exchange-method dh-group1-sha1** command. This prevents the client from connecting to ICX device using SSH with a lesser secured key exchange algorithm.

NOTE
High CPU usage is expected while establishing SSH sessions with diffie-hellman-group14-sha1 key-exchange method.

Enabling empty password logins

By default, empty password logins are not allowed. This means that users with an SSH client are always prompted for a password when they log into the device. To gain access to the device, each user must have a user name and password. Without a user name and password, a user is not granted access.

If you enable empty password logins, users are not prompted for a password when they log in. Any user with an SSH client can log in without being prompted for a password.

To enable empty password logins, enter the following command.

```
device# configure terminal
device(config)# ip ssh permit-empty-passwd yes
```

To disable empty password logins, enter the following command.

```
device# configure terminal
device(config)# ip ssh permit-empty-passwd no
```

Setting the SSH port number

By default, SSH traffic occurs on TCP port 22. You can change this port number. However, if you change the default SSH port number, you must configure SSH clients to connect to the new port. You must also be careful not to assign SSH to a port that is used by another service.

NOTE
If you change the SSH port number, Ruckus recommends that you change it to a port number greater than 1024.

The following command changes the SSH port number to 2200.

```
device# configure terminal
device(config)# ip ssh port 2200
```

Setting the SSH login timeout value

When the SSH server attempts to negotiate a session key and encryption method with a connecting client, it waits a maximum of 120 seconds for a response from the client. If there is no response from the client after 120 seconds, the SSH server disconnects.

You can change this timeout value to from 1 through 120 seconds. For example, to change the timeout value to 60 seconds, enter the following command.

```
device# configure terminal
device(config)# ip ssh timeout 60
```

Designating an interface as the source for all SSH packets

You can designate a loopback interface, virtual interface, or Ethernet port as the source for all SSH packets from the device.

Configuring the maximum idle time for SSH sessions

By default, SSH sessions do not time out. You have the option to set the amount of time an SSH session can be inactive before the Ruckus device closes it. For example, to set the maximum idle time for SSH sessions to 30 minutes, enter the following command.

```
device# configure terminal
device(config)# ip ssh idle-time 30
```

If an established SSH session has no activity for the specified number of minutes, the Ruckus device closes it. An idle time of 0 minutes (the default value) means that SSH sessions never time out. The maximum idle time for SSH sessions is 240 minutes.

SSH rekey exchange

In an SSH2 implementation, if an SSH session is authenticated and established, it remains connected until the user closes it or it is closed after the configured idle time limit. Prolonged usage of the session key negotiated at connection startup poses several security issues and exposes the SSH connection to man-in-middle attacks. To safeguard the SSH connection from security vulnerability, new key exchanges should take place frequently for existing SSH sessions.

SSH rekeying is the process of exchanging the session keys at a configured interval, either in terms of a time limit or a data limit for an SSH session. SSH rekeying is triggered when the maximum number of minutes has been reached or when the maximum data has been reached for the particular session. Rekeying can be initiated by both SSH client and SSH server. While the key exchange renegotiation is taking place, data is not passed through the SSH connection. The algorithm that was used at connection startup is used during rekeying. SSH rekey exchange is supported on OpenSSH-based client version 7.5 or later.

SSH rekey configuration notes

- The encryption method must not be modified during rekey.
- When the rekey configuration is changed, the behavior of the existing session will not have any impact until the next rekey exchange happens for the corresponding session.
- When the rekey exchange occurs, the value of data and the time of the corresponding SSH session will reset to the configured rekey value.
- When rekey is enabled, the existing SSH session will not have the rekey functionality until the rekey exchange occurs from the other side.
- SSH sessions established without the rekey configuration will not have the rekey functionality.
- When the rekey functionality is enabled with few SSH sessions, the corresponding SSH session will not have the rekey functionality until the rekey exchange occurs from the other side.
- Removing the rekey configuration disables the SSH rekey for existing SSH sessions.

Filtering SSH access using ACLs

By default, SSH access is not restricted. You can permit or deny SSH access to the Ruckus device using ACLs. To use ACLs, first create the ACLs you want to use. You can specify a numbered standard IPv4 ACL, a named standard IPv4 ACL, or a named IPv6 access list.

The following example configures IPv4 numbered access list 10 that permits only two specific hosts, denies and logs a third host, and denies all other hosts.

```
device# configure terminal
device(config)# access-list 10 permit host 10.168.144.241
device(config)# access-list 10 deny host 10.168.144.242 log
device(config)# access-list 10 permit host 10.168.144.243
device(config)# access-list 10 deny any
device(config)# ssh access-group 10
```

Terminating an active SSH connection

To terminate one of the active SSH connections, enter the **kill** command followed by the session number as shown in the following example.

```
device# kill ssh 1
```

Refer to the *Ruckus FastIron Command Reference* for more information on the **kill** command.

Displaying SSH information

Up to five SSH connections can be active on the Ruckus device.

Displaying SSH connection information

To display information about SSH connections, enter the **show ip ssh** command.

```
device# show ip ssh
Connection  Version  Encryption  Username  HMAC       Server Hostkey  IP Address
Inbound:
     1      SSH-2    3des-cbc    Raymond   hmac-sha1  ssh-dss         10.120.54.2
Outbound:
     6      SSH-2    aes256-cbc  Steve     hmac-sha1  ssh-dss         10.37.77.15
SSH-v2.0 enabled; hostkey: DSA(1024), RSA(2048)

device# show ip ssh
Connection  Version  Encryption  Username  HMAC       Server Hostkey  IP Address
Inbound:
     1      SSH-2    aes128-ctr    Raymond   hmac-sha1  ssh-dss         10.120.54.2
Outbound:

SSH-v2.0 enabled; hostkey: DSA(1024), RSA(2048)
```

Displaying SSH configuration information

To display SSH configuration information, enter the **show ip ssh config** command.

```
device# show ip ssh config
SSH server                 : Disabled
SSH port                   : tcp\22
```

```
Host Key                    : DSA 1024
Encryption                  : aes256-cbc, aes192-cbc, aes128-cbc, aes256-ctr, aes
                              192-ctr, aes128-ctr, 3des-cbc
Permit empty password       : No
Authentication methods      : Password, Public-key, Interactive
Authentication retries      : 3
Login timeout (seconds)     : 120
Idle timeout (minutes)      : 0
Strict management VRF       : Disabled
SCP                         : Enabled
SSH IPv4 clients            : All
SSH IPv6 clients            : All
SSH IPv4 access-group       :
SSH IPv6 access-group       :
SSH Client Keys             : RSA(0)
Client Rekey                : 0 Minute, 0 KB
Server Rekey                : 0 Minute, 0 KB
```

Displaying additional SSH connection information

The **show who** command also displays information about SSH connections.

```
device# show who
       Console connections:
       Established
       you are connecting to this session
       2 minutes 56 seconds in idle
SSH server status: Enabled
SSH connections (inbound):
1. established, client ip address 10.2.2.1, server hostkey DSA
   1 minutes 15 seconds in idle
2. established, client ip address 10.2.2.2, server hostkey RSA
   2 minutes 25 seconds in idle
SSH connection (outbound):
3. established, server ip address 10.37.77.15, server hostkey RSA
    7 seconds in idle
```

SSH2 client

SSH2 client allows you to connect from a Ruckus ICX device to an SSH2 server, including another Ruckus ICX device that is configured as an SSH2 server. You can start an outbound SSH2 client session while you are connected to the device by any connection method (SSH2, Telnet, console). Ruckus ICX devices support one outbound SSH2 client session at a time.

The supported SSH2 client features are as follows:

- Encryption algorithms, in the order of preference:
  - aes256-ctr
  - aes192-ctr
  - aes128-ctr
  - aes256-cbc
  - aes192-cbc
  - aes128-cbc
  - 3des-cbc
- SSH2 client session authentication algorithms:
  - Password authentication
  - Public Key authentication
- Message Authentication Code (MAC) algorithm: hmac-sha1
- Key exchange algorithm: diffie-hellman-group1-sha1 or diffie-hellman-group14-sha1

- No compression algorithms are supported.
- The client session can be established through either in-band or out-of-band management ports.
- The client session can be established through IPv4 or IPv6 protocol access.
- The client session can be established to a server listening on a non-default SSH port.

Enabling SSH2 client

To use SSH2 client, you must first enable SSH2 server on the device. See SSH2 authentication types on page 82.

When SSH2 server is enabled, you can use SSH client to connect to an SSH server using password authentication.

Configuring SSH2 client public key authentication

To use SSH client for public key authentication, you must generate SSH client authentication keys and export the public key to the SSH servers to which you want to connect.

The following sections describe how to configure SSH client public key authentication:

Generating and deleting a client DSA key pair

To generate a client DSA key pair, enter the following command.

```
device# configure terminal
device(config)# crypto key client generate dsa
```

To delete the DSA host key pair, enter the following command.

```
device# configure terminal
device(config)# crypto key client zeroize dsa
```

Generating and deleting a client RSA key pair

To generate a client RSA key pair with the default modulus size of 1024 bits, enter the following command.

```
device# configure terminal
device(config)# crypto key client generate rsa
```

To generate a client RSA key pair with a modulus size of 2048 bits, enter the following command.

```
device# configure terminal
device(config)# crypto key client generate rsa modulus 2048
```

To delete the RSA host key pair, enter the following command.

```
device# configure terminal
device(config)# crypto key client zeroize rsa
```

NOTE
The **crypto key client zeroize** command without additional parameters removes all RSA and DSA keys from flash memory, thereby disabling SSH.

Exporting client public keys

Client public keys are stored in the following files in flash memory:

- A DSA key is stored in the file `$$sshdsapub.key`.
- An RSA key is stored in the file `$$sshrsapub.key`.

To copy key files to a TFTP server, you can use the **copy flash tftp** command.

You must copy the public key to the SSH server. If the SSH server is a Ruckus ICX device, see the section Importing authorized public keys into the device on page 85.

Establishing an SSH2 client connection

To start an SSH2 client connection to an SSH2 server using password authentication, enter the **ssh** command followed by the IPv4 or IPv6 address of the host as shown in the following example.

```
device# ssh 10.10.10.2
```

The example establishes a connection with the SSH host at IPv4 address 10.10.10.2.

To start an SSH2 client connection to an SSH2 server using public key authentication, enter the ssh command, the IP address of the host, the keyword public-key and the type of public key to be used as shown in the following example.

```
device# ssh 10.10.10.2 public-key dsa
```

The example establishes an SSH connection with the host at IPv4 address 10.10.10.2 using DSA public-key authentication.

For more information on the **ssh** command, refer to the *Ruckus FastIron Command Reference*.

Displaying SSH2 client information

For information about displaying SSH2 client information, see the following sections:

- Displaying SSH connection information on page 90
- Displaying additional SSH connection information on page 91

SCP client support

SCP client

Secure copy (SCP) supports file transfer between local and a remote hosts. It combines the file-transfer element of BSD remote copy (RCP) with the authentication and encryption provided by the Secure shell (SSH) protocol.

The SCP client feature on Ruckus ICX devices helps to transfer files to and from the SCP server and maintains the confidentiality of the data being transferred by blocking packet sniffers from extracting valuable information from the data packets. You can use SCP client to do the following:

- Download a boot file, FastIron application image file, signature file, license file, startup configuration file, or running configuration from an SCP server
- Upload a FastIron application image file, startup configuration file, or running configuration to an SCP server
- Upgrade the PoE firmware by downloading a file from an SCP server

SCP client uploads the file to the SCP server (that is, the SSH server) by providing files to be uploaded. You can specify file attributes, such as permissions and time-stamps as part of file data when you use SCP client to upload files. It supports the same copy features as the timestamps, TFTP client feature on FastIron devices, but the SSH2 protocol secures data transfer.

SCP client support limitations

SCP client sessions are limited by file size and by whether other SCP client sessions are running and by whether SC server sessions are in progress.

The following limitations apply to SCP client sessions:

- An SCP copy of the running or startup configuration file from a Ruckus ICX device to Linux WS 4 or 5 may fail if the configuration size is less than 700 bytes.
- Only one SCP client session is supported at a time.
- An SCP client session cannot be initiated if an SCP server session is in progress.
- An SSH client outbound session cannot be initiated if an SCP client session is in progress from the same terminal.
- Uploading and downloading public or private key files is not supported.
- Downloading signature files is not supported.

- When transferring files between devices under test (DUTs), the following limitations apply:
  - When using a binary image copy to transfer files between DUTs, you should configure the **flash:primary** keyword rather than the **primary** keyword because the SCP server does not support remote-filename aliases. See the description of the **copy scp flash** or the **copy flash scp** command for more information.
  - Be sure to download the compatible configurations when you transfer startup configuration or running configuration files copy between DUTs because the overwrite option is restricted.
  - Copying power over Ethernet (POE) firmware between two DUTs is not supported.
  - During Image copy between two mixed stacking units, KX image copy is not supported and cant upload the KX image from mixed stacking to Linux or Windows servers.
  - Bootrom image copy between two DUTs is not supported.
  - License copy between two DUTs is not supported.
  - Manifest file copy between two DUTs is not supported.

Supported SCP client configurations

SCP client automatically uses the authentication methods, encryption algorithm, and data compression level configured for SSH.

For example, if password authentication is enabled for SSH, you are prompted for a user name and password before SCP allows a file to be transferred.

The following conditions also apply:

- SCP is enabled by default and can be enabled or disabled using the **ip ssh scp disable | enable** command.
- If SSH is disabled, SCP is disabled automatically.
- The SCP client session uses one SSH outbound client session.
- Because the SCP client internally uses the SSH2 client for creating outbound SSH sessions from the device, all configurations related to the SSH2 client are required for SCP client support, as described here:
  - The SSH2 server on the device must be enabled by creating an SSH server DSA or RSA key pair; otherwise, the SSH2 client cannot be used.
  - You can use the **crypto key client { generate | zeroize } dsa** command to generate or delete an SSH-client-DSA key pair. The SSH-client-DSA public key is stored in the file - $$sshdsapub.key.
  - You can use the **crypto key client generate rsa [modulus 1024 | 2048]** command to generate an SSH-client-RSA key pair. The SSH-client-RSA public key is stored in the file $$sshrsapub.key.
  - You can use the **crypto key client zeroize rsa** command to delete an SSH-client-RSA key pair.

Beginning with 8.0.30d release, the SCP file transfer speed over high latency connections is increased.

Downloading an image from an SCP server

Securely download image files from a secure copy (SCP) server.

Copy an image from the SCP server to a device as shown in the following example.

```
device# copy scp flash 10.20.1.1 SPR08040.bin primary
device# copy scp flash 10.20.1.1 SPR08040.bin secondary
```

The example copies a primary and secondary image from the SCP server with the IP address 10.20.1.1.

For more information on the **copy scp flash** command, refer to the *Ruckus FastIron Command Reference*.

Uploading an image to an SCP server

To securely upload image files to a secure copy (SCP) server, copy an image from a device to the SCP server.

```
device# copy flash scp 10.20.1.1 SPR08040.bin primary
device# copy flash scp 10.20.1.1 SPR08040.bin secondary
```

Uploading configuration files to an SCP server

To securely upload startup and running configuration files to a secure copy (SCP) server.

1. Copy a startup configuration file to the SCP server.

   ```
   device# copy startup-config  scp 10.20.1.1 icx-74-startup
   ```

   The startup configuration file is uploaded to the SCP server and you are notified when the transfer is complete.

   ```
   user name: name
   Password:
   Connecting to remote host......

   Sending data (8192 bytes per dot)
   .

   SCP transfer from device completed

   SYSLOG: <14>2014 Apr  1 14:34:16 ICX-74-CC SCP transfer from device completed

   Connection Closed
   ```

2. Copy a running configuration file to the SCP server.

   ```
   device# copy running-config  scp 10.20.1.1 icx-74-run
   ```

Downloading configuration files from an SCP server

To securely download startup and running configuration files from a secure copy (SCP) server to a device.

1. Copy a startup configuration file from the SCP server.

   ```
   device# copy scp startup-config 10.20.1.1 icx-74-startup
   ```

2. Copy a running configuration file from the SCP server.

   ```
   device# copy scp running-config 10.20.1.1 icx-74-run
   ```

Copying an image between devices

Securely copy image files between FastIron devices.

Copy an image between devices.

```
device# copy flash scp 10.20.66.15 flash:sec:SPR08040.bin primary
device# copy scp flash 10.20.66.15 flash:secondary primary
```

Secure copy with SSH2

Secure Copy (SCP) uses security built into SSH to transfer image and configuration files to and from the device. SCP automatically uses the authentication methods, encryption algorithm, and data compression level configured for SSH. For example, if password authentication is enabled for SSH, the user is prompted for a user name and password before SCP allows a file to be transferred. No additional configuration is required for SCP on top of SSH.

You can use SCP to copy files on the Ruckus device, including the startup configuration and running configuration files, to or from an SCP-enabled remote host.

Enabling and disabling SCP

SCP is enabled by default and can be disabled.

NOTE
If you disable SSH, SCP is also disabled.

To disable SCP, enter the following command.

```
device# configure terminal
device(config)# ip ssh scp disable
```

To re-enable SCP, enter the following command.

```
device# configure terminal
device(config)# ip ssh scp enable
```

Secure copy configuration notes

- When using SCP, enter the **scp** commands on the SCP-enabled client, rather than the console on the Ruckus device.
- You may see the error "protocol error: filename does not match request" when attempting to perform an SCP copy of the following types of files: running-config, startup-config, ICX software image. This may indicate that you did not include the file extension in the source filename. Either re-enter the command and include the filename extension, or disable SCP strict filename checking feature using the -t option.
- Certain SCP client options, including -p and -r, are ignored by the SCP server on the Ruckus device. If an option is ignored, the client is notified.
- An SCP AES copy of the running or start configuration file from the Ruckus device to Linux WS 4 or 5 may fail if the configuration size is less than 700 bytes. To work around this issue, use PuTTY to copy the file.
- SCP does not support running config overwite except acl configuration.

Example file transfers using SCP

The following are examples of using SCP to transfer files to and from a Ruckus device.

Copying a file to the running config

To copy a configuration file from an SCP-enabled client to the running configuration on a Ruckus ICX, enter the **scp** command followed by the location of the source file, the name of the user logging in, the IP address of the Ruckus ICX device, and the destination filename (runConfig) preceded by a colon and no space as shown in the following example.

```
C:\> scp c:\cfg\default.cfg terry@10.168.1.50:runConfig
```

The example copies the configuration file located at c:\cfg\default.cfg to the running configuration file (runConfig) on a Ruckus ICX device with the IP address 10.168.1.50 and logs in as the user terry.

NOTE
If password authentication is enabled for SSH, the user is prompted for the password of the user (terry in the example) before the file transfer takes place.

Copying a file to the startup config

To copy the configuration file from an SCP-enabled device to the startup configuration file of a Ruckus ICX device, enter the **scp** command, the source file path, the username and IP address of the Ruckus ICX device, followed by a colon and no space, and the destination filename as shown in the following example.

```
C:\> scp c:\cfg\default.cfg terry@10.168.1.50:startConfig
```

The example copies the configuration file default.cfg to the Ruckus ICX device with the IP address 10.168.1.50, logs in the user terry, and stores the file as the startup configuration file startConfig.

NOTE
If password authentication is enabled for SSH, the user is prompted for the password of the user (terry in the example) before the file transfer takes place.

Copying the running config file to an SCP-enabled client

To copy the running config file from the Ruckus ICX device to an SCP-enabled client, enter the scp command followed by the username, the IP address of the Ruckus ICX device, a colon, no space and the ICX running configuration filename (runConfig), and the target location on the client as shown in the following example.

```
C:\> scp terry@10.168.1.50:runConfig c:\cfg\brcdrun.cfg
```

The example copies the running configuration file (runConfig) from the Ruckus ICX device referenced by username and IP address (terry@10.168.1.50) to a file called c:\cfg\fdryrun.cfg on an SCP-enabled client.

Copying the startup config file to an SCP-enabled client

To copy the startup configuration file from the Ruckus device to an SCP-enabled client, enter the **scp** command, the username and IP address of the Ruckus device, and the full startup config filename (preceded by a colon and no space), followed by the target filename as shown in the following example.

```
C:\> scp terry@10.168.1.50:startConfig.cfg c:\cfg\brcdstart.cfg
```

The example copies the startup configuration file (startConfig.cfg) from the Ruckus device, identified by the username and IP address terry@10.168.1.50, to the destination file c:\cfg\brcdstart.cfg on an SCP-enabled client.

Copying a software image file to flash memory

To copy a software image file from an SCP-enabled client to the primary flash of a Ruckus ICX device, enter the **scp** command, the image filename, the ICX username, and the IP address of the Ruckus ICX device. On the same command line, with no spaces, enter a colon and the keyword **flash**, another colon and the keyword **primary**.

```
C:\> scp SPR08040.bin terry@10.168.1.50:flash:primary
```

The previous example copies the image file for the FastIron 08.0.40 software release to the a Ruckus ICX device represented by the username terry and the IP address 10.168.1.50. The image file is copied to primary flash.

The following example is an equivalent command that explicitly includes the identical image filename as the destination filename.

```
C:\>scp SPR08040.bin terry@10.168.1.50:flash:pri:SPR08040.bin
```

Use the same command structure to copy a software image file from an SCP-enabled client to the secondary flash on a Ruckus ICX device, but replace the keyword **primary** with the keyword **secondary** as shown in the following example.

```
C:\> scp SPR08040.bin terry@10.168.1.50:flash:secondary
```

The following example is an equivalent command explicitly includes the identical image filename as the destination filename. Notice that the keyword **sec** is used in the command string.

```
c:\> scp SPR08040.bin user@10.168.1.50:flash:sec:SPR08040.bin
```

NOTE
After the copy operation is completed at the host, you do not get the command prompt back because the switch is synchronizing the image to flash. To ensure that you have successfully copied the file, issue the **show flash** command. If the copy operation is not complete, the **show flash** command output shows the partition (primary or secondary) as EMPTY.

NOTE
The Ruckus ICX device supports only one SCP copy session at a time.

Copying a Software Image file from flash memory

To copy a software image file from the primary flash of a a Ruckus ICX device to an SCP-enabled client, enter the scp command followed by the ICX username and IP address and, without spaces, the key phrase **:flash:primary** followed by a space and the image filename as shown in the following example.

```
C:\> scp terry@10.168.1.50:flash:primary SPR08040.bin
```

The example copies the FastIron release 08.0.40 image file from the primary flash of the username and Ruckus ICX device at the specified IP address (terry@10.168.1.50) to an SCP-enabled client.

The command to copy an image from the secondary flash of a Ruckus ICX device to an SCP-enabled client replaces the keyword **primary** with the keyword **secondary** as shown in the following example.

```
C:\> scp terry@10.168.1.50:flash:secondary SPR08040.bin
```

The second example copies the FastIron release 08.0.40 image file from the secondary flash of the same ICX user and device to an SCP-enabled client.

Importing a digital certificate using SCP

The **scp** command can be used to import a certificate file when TFTP access is unavailable or not permitted and the command has an equivalent functionality to the **ip ssl certificate-data-file tftp** command.

To import a digital certificate using SCP, enter the **scp** command followed by the filename of the certificate to be imported, a valid username on the host, the host IP address from which the file is to be imported, and the license to be imported as shown in the following example.

```
C:\> scp certfile user@10.168.89.210:sslCert
```

The example imports the digital file named certfile from a valid named user on the host at the specified host address (user@10.168.89.210) and stores it on the Ruckus ICX device with the filename sslCert.

Importing an RSA private key

The **scp** command can be used to import a private key when TFTP access is unavailable or not permitted and the command has an equivalent functionality to the **ip ssl private-key-file tftp** command.

To import an RSA private key from a client using SCP, enter the **scp** command followed by the filename of the key to be imported, a valid username on the host, the host IP address from which the key is to be imported, and the destination filename sslPrivKey as shown in the following example.

```
C:\> scp keyfile user@10.168.9.210:sslPrivKey
```

The example imports the private RSA key named keyfile from "user" on the server with the IP address 10.168.9.210 and stores it in the designated RSA key file sslPrivKey on the Ruckus ICX device.

Importing a DSA or RSA public key

The **scp** command can be used when TFTP access is unavailable or not permitted and the command has an equivalent function to the **ip ssh pub-key-file tftp** command. For more information on the **ip ssh pub-key-file tftp** command, refer to Importing authorized public keys into the device on page 85.

To import a DSA or RSA public key from a client using SCP, enter the **scp** command followed by the filename of the public DSA or RSA key to be imported, the name of a valid user on the server and the server IP address from which the key is to be imported, and the filename sshPubKey, where the key is stored on the Ruckus ICX device.

```
C:\> scp pkeys.txt user@10.168.1.234:sshPubKey
```

The example imports the private key file pkeys.txt from a valid user on the server at IP address 10.168.1.234 (user@10.168.1.234) and stores it in the key file sshPubKey on the Ruckus ICX device.

Copying license files

To use SCP to copy the license files from a client to a Ruckus ICX stack unit, enter the **scp** command followed by the name of the license file to be imported, the username on the server, the server IP address, the keyword **license**, and the number of the Ruckus ICX stack unit to which the license is to be copied.

```
C:\> scp license.xml user@10.168.1.234:license:3
```

The example copies the license file license.xml from a designated user on the server (user@10.168.1.234) to Ruckus ICX stack unit 3.

ACLs

Layer 3 ACL overview

Layer 3 (IPv4 and IPv6) access control lists (ACLs) permit or deny packets according to rules included in the ACLs.

When a 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 packet.

ACLs include the following benefits:

- Providing security and traffic management.
- Monitoring network and user traffic.
- Saving network resources by classifying traffic.
- Protecting against denial of service (DoS) attacks.
- Reducing debug output.

Because applied ACLs are programmed into the Content Addressable Memory (CAM), packets are permitted or denied in the hardware, without sending the packets to the CPU for processing.

Layer 3 ACLs are implemented using the following flow:

1. Create the ACL, using the **ip access-list** or **ipv6 access-list** command.
2. Define permit and deny rules, using the [**sequence** *seq-num*] { **deny** | **permit** } command.
3. Apply the ACL to one or more interfaces, using the relevant command:
   - IPv4: **ip access-group**
   - IPv6: **ipv6 traffic-filter**

Layer 3 ACLs are supported on the following interface types:

- 1 Gigabit Ethernet (1-GbE) ports
- 10 Gigabit Ethernet (10-GbE) ports
- 40 Gigabit Ethernet (40-GbE) ports
- 100 Gigabit Ethernet (100-GbE) ports
- Trunk groups
- Virtual routing interfaces

Although you can assign a number to IPv4 ACLs, named ACLs are supported for both IPv4 and IPv6 ACLs. Named ACLs must begin with an alphabetical character, can contain up to 255 characters and numbers, and must be unique among both IPv4 and IPv6 ACLs.

NOTE
For Layer 2 filtering, refer to Defining MAC Address Filters on page 173.

NOTE
For ACLs under Flexible Authentication, refer to Dynamic ACLs in authentication on page 211.

ACL and rule limits

For the various ACL types, there are software limits to the number of ACLs supported. The maximum number of ACL rules supported varies with the device. The following table provides information on ACL scaling.

TABLE 13 ACL and rule limits

Security feature	ICX 7150	ICX 7250	ICX 7450	ICX 7650	ICX 7750	ICX 7850
Single ACL-ID						
IPv4 ACLs - Numbered (Standard)	99	99	99	99	99	99
IPv4 ACLs - Numbered (Extended)	100	100	100	100	100	100
IPv4 ACLs - Named (Standard)	600	600	600	600	600	600
IPv4 ACLs - Named (Extended)	600	600	600	600	600	600
IPv6 ACLs - Named	600	600	600	600	600	600
# of IPv4 rules possible to configure in 1 ACL - ID before binding to interface	2,000	2,000	2,000	2,000	2,000	2,000
# of IPv6 rules possible to configure in 1 ACL - ID before binding to interface	2,000	2,000	2,000	2,000	2,000	2,000
Max IPv4 ingress HW TCAM entries possible in one device including default rules	512	2,816	2,560	4,096	2,048	1,536
Max IPv6 ingress HW TCAM entries possible in 1 device including default rules	256	1,408	1,280	2,048	1,024	1,536
Max egress HW TCAM entries possible in 1 device including default rules	256 (IPv4- 128, IPv6-128)	512 (IPv4- 256, IPv6-256)	512 (IPv4- 256, IPv6-256)	512 (IPv4- 256, IPv6-256)	512 (IPv4- 256, IPv6-256)	1024 (IPv4- 512, IPv6-512)

TABLE 13 ACL and rule limits (continued)

Security feature	ICX 7150	ICX 7250	ICX 7450	ICX 7650	ICX 7750	ICX 7850
Multiple ACL-IDs						
Max rules configuration in overall system (cumulative of all rules) Configure only, rules not bound to HW	8,192	8,192	8,192	8,192	8,192	8,192

Default ACL action

The default action when no ACLs are configured on a device is to permit all traffic. However, once you configure an ACL and apply it to a port, the default action for that port is to deny all traffic that is not explicitly permitted on the port. Given these defaults, follow these guidelines for configuring and applying ACLs:

- If you want to tightly control access, configure ACLs consisting of permit entries for the access you want to permit. The ACLs implicitly deny all other access.
- If you want to secure access in environments with many users, you may want to configure ACLs that consist of explicit deny entries, and then add an entry to permit all access to the end of each ACL. The software permits packets that are not denied by the deny entries.

Because by default traffic is restricted unless explicitly allowed once an ACL is applied to an interface, an ACL can block packets switched between members of the same VLAN or subnet over a virtual interface with an ACL applied to it. To overcome this issue, you must add explicit rules to the ACL allowing traffic from the sources you want to include.

Consider the following example.

```
Vlan 200
Untagged e port 1/1/1 e 1/1/2
Router-interface ve 200

Interface ve 200
Ip address 192.168.1.1/24


Host1 (192.168.1.10/24)  connected to 1/1/1
and Host2 (192.168.1.11/24) connected to 1/1/2

Access-list 101
permit ip host 1.1.1.1 host 2.2.2.2

Access-list 102
permit ip host 1.1.1.1 host 2.2.2.2
permit ip 192.168.1.0/24
```

If Access-list 101 is applied to virtual interface 200, the hosts in VLAN 200 are NOT able to communicate with each other. This is because ACL 101 includes an implicit "deny" action at the end by default.

However, if Access-list 102 is applied to virtual interface 200, Host1 and Host 2 ARE able to communicate with each other. This is because the second statement of the ACL explicitly permits traffic to and from the IP subnet 192.168.1.0/24, to which both hosts belong.

How hardware-based ACLs work

When you bind an ACL to inbound or outbound traffic on an interface, the device programs the Layer 4 CAM with the ACL. Permit and deny rules are programmed. Most ACL rules require one Layer 4 CAM entry. However, ACL rules that match on more than one TCP or UDP application port may require several CAM entries. The Layer 4 CAM entries for ACLs do not age out. They remain in the CAM until you remove the ACL:

- If a packet received on the interface matches an ACL rule in the Layer 4 CAM, the device permits or denies the packet according to the ACL.
- If a packet does not match an ACL rule, the packet is dropped because the default action on an interface that has ACLs is to deny the packet.

How fragmented packets are processed

Fragments are processed by hardware-based ACLs by default in the following manner:

- The first fragment of a packet is permitted or denied using the ACLs. The first fragment is handled the same way as non-fragmented packets, because the first fragment contains the Layer 4 source and destination application port numbers. The device uses the Layer 4 CAM entry if one is programmed, or applies the interface's ACL entries to the packet and permits or denies the packet according to the first matching ACL.
- Other fragments of the same packet are subject to a rule only if there is no Layer 4 information in the rule or in any preceding rules.

The fragments are forwarded even if the first fragment, which contains the Layer 4 information, was denied. Generally, denying the first fragment of a packet is sufficient, because a transaction cannot be completed without the entire packet.

For tighter control, you can configure the port to drop all packet fragments. Refer to Enabling strict control of ACL filtering of fragmented packets on page 113.

IPv4 ACLs

IPv4 ACLs permit or deny IPv4 packets according to rules included in the ACLs.

Regarding the range of filtering options, there are two types of IPv4 ACLs:

- Standard ACLs: Permit or deny traffic according to source address only.
- Extended ACLs: Permit or deny 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 parameters:
  - Port name or number
  - Protocol (for example, TCP or UDP)
  - TCP flags

Regarding ACL naming, there are two types of IPv4 ACLs:

- Numbered ACLs:
  - You can assign numbers 1 through 99 to standard numbered IPv4 ACLs.
  - You can assign numbers 100 through 199 to extended numbered IPv4 ACLs.
- Named ACLs, which must begin with an alphabetical character.

IPv4 ACL configuration guidelines

- The following ACL features and options are NOT supported on the FastIron devices:
  - Flow-based ACLs
  - Layer 2 ACLs
  - Applying an ACL on a device that has Super Aggregated VLANs (SAVs) enabled.
- On ICX 7850 devices only, configuration of egress ACLs is blocked on any virtual interface with an associated VLAN that contains an untagged port.
- ACLs are not supported on Group VEs, even though the CLI contains commands for this action.
- Inbound ACLs apply to all traffic, including management traffic. By default, outbound ACLs are not applied to traffic generated by the CPU. To enable the application of outbound ACLs to CPU traffic, use the **enable egress-acl-on-cpu-traffic** command. Refer to Applying egress ACLs to control (CPU) traffic on page 110 for details.
- Hardware-based ACLs support only one ACL per port. The ACL can contain multiple entries (rules). For example, hardware-based ACLs do not support ACLs 101 and 102 on port 1, but hardware-based ACLs do support ACL 101 containing multiple entries. If a user tries to apply a second ACL on a port, the second ACL will replace the current ACL.
- Inbound ACLs and outbound ACLs can coexist. When an inbound ACL and an outbound ACL are configured on the same port, the outbound ACL is applied only on outgoing traffic.
- By default, the first fragment of a fragmented packet received by a FastIron device is permitted or denied using the ACLs, but subsequent fragments of the same packet are forwarded in hardware. Generally, denying the first fragment of a packet is sufficient, because a transaction cannot be completed without the entire packet.
- ACLs are supported on member ports of a VLAN on which DHCP snooping and Dynamic ARP Inspection (DAI) are enabled. Also, IP source guard and ACLs are supported together on the same port, as long as both features are configured at the port level or per-port-per-VLAN level. IP source guard and ACLs are not supported on the same port if one is configured at the port level and the other is configured at the per-port-per-VLAN level.
- (Router image only) When an ACL is applied on a VLAN physical port, if you need to change VLAN membership, you first need to remove the ACL.
- Ingress MAC filters can be applied to the same port as an outbound ACL.
- A DoS attack configuration on a port will only apply on the ingress traffic.
- Outbound ACLs cannot be configured through a RADIUS server as dynamic or user-based ACLs. However, outbound ACLs can still be configured with MAC authentication or 802.1X authentication enabled, as they are configured in different directions.
- On all platforms—in the router image—ACL support for switched inbound and outbound traffic is enabled by default.
- You can apply an ACL to a port that has TCP SYN protection or ICMP smurf protection, or both, enabled.
- When forming LAGs, make sure that no ACLs are currently applied to the relevant physical interfaces.
- When using host names in ACL filter configurations, please note that a TCAM entry will be programmed with the first IP address resolved through DNS service. Any further updates to DNS entries do not automatically update TCAM.
- It is not possible to configure conflicting ACL filters. When configuring an ACL filter, if the sequence number already exists, or the sequence number is not specified explicitly and all the filter parameters match with an existing filter but the action does not match, the following error appears: *Error: ACL operation failed for ACL ipv4-test1 since following conflicting filter entry already exists. Please use explicit sequence number to override this error.*

Creating a standard numbered IPv4 ACL

A standard ACL permits or denies traffic according to source address only. The following steps create standard ACLs identified by integers 1 through 99.

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 12
```

3. For each rule, enter the [**sequence** *seq-num*] { **deny** | **permit** } command, specifying the needed parameters.

```
device(config-std-nacl)# deny host 10.157.22.26 log
device(config-std-nacl)# deny 10.157.29.12 log
device(config-std-nacl)# deny host IPHost1 log mirror
device(config-std-nacl)# permit any
```

4. Apply the ACL you created to the needed interfaces.

```
device(config-std-nacl)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# ip access-group 12 in
```

The following example configures an ACL to deny packets from three source IP addresses from being received on port 1/1/1. The last rule permits all packets not explicitly denied by the first three ACL entries. (Otherwise, the implicit action is "deny".) Logging is enabled on this port and configured for the deny rules.

```
device# configure terminal
device(config)# ip access-list standard 12
device(config-std-nacl)# deny host 10.157.22.26 log
device(config-std-nacl)# deny 10.157.29.12 log
device(config-std-nacl)# deny host IPHost1 log mirror
device(config-std-nacl)# permit any
device(config-std-nacl)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# acl-logging
device(config-if-e1000-1/1/1)# ip access-group 12 in
```

The following example is the result of entering **show access-list** for the previous ACL. The results indicate automatic assignment of sequence numbers.

```
device# show access-list 12
Standard IP access list 12 : 4 entry
10: deny host 10.157.22.26 log
20: deny 10.157.29.12 log
30: deny host IPHost1 log
40: permit any
```

Creating a standard named IPv4 ACL

A standard named ACL permits or denies traffic according to source address only. The following steps create standard ACLs identified by a name beginning with an alphabetical character.

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 ip_stan_test
```

3. For each rule, enter the [**sequence** *seq-num*] { **deny** | **permit** } command, specifying the needed parameters.

```
device(config-std-nacl)# deny host 10.157.22.26 log
device(config-std-nacl)# deny 10.157.29.12 log
device(config-std-nacl)# deny host IPHost1 log
device(config-std-nacl)# permit any
```

4. Apply the ACL you created to the needed interfaces.

```
device(config-std-nacl)# exit
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# ip access-group ip_stan_test in
```

The following example configures an ACL to deny packets from three source IP addresses from being received on port 1/1/1. The last rule permits all packets not explicitly denied by the first three ACL entries. (Otherwise, the implicit action is "deny".) Logging is enabled on this port and configured for the deny rules.

```
device# configure terminal
device(config)# ip access-list standard ip_stan_test
device(config-std-nacl)# deny host 10.157.22.26 log
device(config-std-nacl)# deny 10.157.29.12 log
device(config-std-nacl)# deny host IPHost1 log
device(config-std-nacl)# permit any
device(config-std-nacl)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# acl-logging
device(config-if-e1000-1/1/1)# ip access-group ip_stan_test in
```

The following example is the result of entering **show access-list** for the previous ACL. The results indicate automatic assignment of sequence numbers.

```
device# show access-list named-acl ip_stan_test
Standard IP access list ip_stan_1 : 4 entry
10: deny host 10.157.22.26 log
20: deny 10.157.29.12 log
30: deny host IPHost1 log
40: permit any
```

Creating an extended numbered IPv4 ACL

Extended ACLs permit or deny traffic according to source and destination addresses, port protocol, and other IPv4 frame content. The following steps create extended ACLs identified by integers 100 through 199.

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 112
```

3. For each rule, enter the [**sequence** *seq-num*] { **deny** | **permit** } command, specifying the needed parameters.

```
device(config-ext-nacl)# deny tcp host 10.157.22.26 any eq telnet log
device(config-ext-nacl)# permit ip any any
```

4. Apply the ACL you created to the needed interfaces.

```
device(config-ext-nacl)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# ip access-group 112 in
```

The following example includes remarks preceding each rule.

```
device# configure terminal
device(config)#ip access-list extended 115
device(config-ext-nacl)# remark Permits ICMP traffic from 10.157.22.x to 10.157.21.x:
device(config-ext-nacl)# permit icmp 10.157.22.0/24 10.157.21.0/24
device(config-ext-nacl)# remark Denies IGMP traffic from "rkwong" to 10.157.21.x:
device(config-ext-nacl)# deny igmp host rkwong 10.157.21.0/24 log
device(config-ext-nacl)# remark Denies IGRP traffic from "rkwong" to 10.157.21.x:
device(config-ext-nacl)# deny igrp 10.157.21.0/24 host rkwong log
device(config-ext-nacl)# remark Denies IPv4 traffic from 10.157.21.100 to 10.157.22.1, with logging:
device(config-ext-nacl)# deny ip host 10.157.21.100 host 10.157.22.1 log
device(config-ext-nacl)# remark Denies all OSPF traffic, with logging:
device(config-ext-nacl)# deny ospf any any log
device(config-ext-nacl)# remark Permits traffic not explicitly denied by the previous rules:
device(config-ext-nacl)# permit ip any any
```

Creating an extended named IPv4 ACL

Extended ACLs permit or deny traffic according to source and destination addresses, port protocol, and other IPv4 frame content. The following steps create extended ACLs identified by a name beginning with an alphabetical character.

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 ip_ext_test
   ```

3. For each rule, enter the [**sequence** *seq-num*] { **deny** | **permit** } command, specifying the needed parameters.

   ```
   device(config-ext-nacl)# deny tcp host 10.157.22.26 any eq telnet log
   device(config-ext-nacl)# permit ip any any
   ```

4. Apply the ACL you created to the needed interfaces.

   ```
   device(config-ext-nacl)# interface ethernet 1/1/2
   device(config-if-e1000-1/1/2)# ip access-group ip_ext_test in
   ```

The following example includes remarks preceding each rule.

```
device# configure terminal
device(config)# ip access-list extended ip_ext_test
device(config-ext-nacl)# remark Permits ICMP traffic from 10.157.22.x to 10.157.21.x:
device(config-ext-nacl)# permit icmp 10.157.22.0/24 10.157.21.0/24
device(config-ext-nacl)# remark Denies IGMP traffic from "rkwong" to 10.157.21.x:
device(config-ext-nacl)# deny igmp host rkwong 10.157.21.0/24 log
device(config-ext-nacl)# remark Denies IGRP traffic from "rkwong" to 10.157.21.x:
device(config-ext-nacl)# deny igrp 10.157.21.0/24 host rkwong log
device(config-ext-nacl)# remark Denies IPv4 traffic from 10.157.21.100 to 10.157.22.1, with logging:
device(config-ext-nacl)# deny ip host 10.157.21.100 host 10.157.22.1 log
device(config-ext-nacl)# remark Denies all OSPF traffic, with logging:
device(config-ext-nacl)# deny ospf any any log
device(config-ext-nacl)# remark Permits traffic not explicitly denied by the previous rules:
device(config-ext-nacl)# permit ip any any
```

Applying egress ACLs to control (CPU) traffic

With the exceptions described in this section, outbound ACLs are not by default applied to traffic generated by the CPU. To apply outbound ACLs to CPU traffic, use the **enable egress-acl-on-cpu-traffic** command.

CPU traffic on ICX 7150 and ICX 7850 stack members and standby-controllers is, by default, subjected to outbound ACL filtering. In contrast, ICX 7150 and ICX 7850 standalone devices and stack active-controllers have the same default behavior as other platforms and do not apply outbound ACLs to CPU traffic unless the **enable egress-acl-on-cpu-traffic** command is configured.

Preserving user input for ACL TCP/UDP port numbers

ACL implementations automatically display the TCP/UDP port name instead of the port number, regardless of user preference, unless the device is configured to preserve user input. When the option to preserve user input is enabled, the system will display either the port name or the number.

To enable this feature, enter the **ip preserve-ACL-user-input-format** command.

```
device(config)# ip preserve-ACL-user-input-format
```

The following example shows how this feature works for a TCP port (this feature works the same way for UDP ports). In this example, the user identifies the TCP port by number (80) when configuring ACL group 140. However, **show ip access-lists 140** reverts to the port name for the TCP port (http in this example). After the user issues the **ip preserve-ACL-user-input-format** command, **show ip access-lists 140** displays either the TCP port number or name, depending on how it was configured by the user.

```
device(config)# ip access-list extended 140
device(config-ext nacl)# permit tcp any any eq 80
device(config-ext nacl)# permit tcp any any eq ftp
device(config-ext nacl)# show ip access-lists 140
Extended IP access list 140
permit tcp any any eq http
permit tcp any any eq ftp
device((config-ext nacl)# permit tcp any any eq 80
device(config-ext nacl)# permit tcp any any eq ftp
device(config-ext nacl)# show ip access-lists 140
Extended IP access list 140
permit tcp any any eq http
permit tcp any any eq ftp
device(config-ext nacl)# ip preserve-ACL-user-input-format
device(config-ext nacl)# show ip access-lists 140
Extended IP access list 140
permit tcp any any eq 80
permit tcp any any eq ftp
```

ACL comment text management

ACL comment text describes entries in an ACL, appearing in the output of show commands that display ACL information.

Adding a comment to an entry in a numbered ACL

To add comments to entries in a numbered ACL, enter commands such as the following:

```
device(config)# ip access-list extended 100
device(config-ext-nacl)# remark The following line permits TCP packets
device(config-ext-nacl)# permit tcp 192.168.4.40/24 2.2.2.2/24
device(config-ext-nacl)# remark The following permits UDP packets
device(config-ext-nacl)# permit udp 192.168.2.52/24 2.2.2.2/24
device(config-ext-nacl)# deny ip any any
```

Adding a comment to an entry in a named ACL

To add comments to entries in a named ACL, enter commands such as the following.

```
device(config)# ip access-list extended TCP/UDP
device(config-ext-nacl)# remark The following line permits TCP packets
device(config-ext-nacl)# permit tcp 192.168.4.40/24 2.2.2.2/24
device(config-ext-nacl)# remark The following permits UDP packets
device(config-ext-nacl)# permit udp 192.168.2.52/24 2.2.2.2/24
device(config-ext-nacl)# deny ip any any
```

Deleting a comment from an ACL entry

To delete a comment from an ACL entry, enter commands such as the following:

```
device(config)# ip access-list standard 99
device(config-std-nacl)# no remark The following line permits TCP packets
```

Viewing comments in an ACL

You can use the following commands to display comments for ACLs:

- **show running-config**
- **show access-list**
- **show access-list named-acl**
- **show ip access-list**

The following example shows the comment text for a numbered ACL, ACL 100, in a **show running-config** display.

```
device# show running-config
...
Extended IP access list 100 remark The following line permits TCP packets
Extended IP access list 100 permit tcp 192.168.4.40/24 2.2.2.2/24
Extended IP access list 100 remark The following line permits UDP packets
Extended IP access list 100 permit udp 192.168.2.52/24 2.2.2.2/24
Extended IP access list 100 deny ip any any
```

The following example shows the comment text for an ACL in a **show access-list** display.

```
device# show access-list 100
IP access list rate-limit 100 aaaa.bbbb.cccc
Extended IP access list TCP/UDP (Total flows: N/A, Total packets: N/A)
ACL Remark: The following line permits TCP packets
permit tcp 0.0.0.40 255.255.255.0 0.0.0.2 255.255.255.0 (Flows: N/A, Packets: N/A)
ACL Remark: The following line permits UDP packets
permit udp 0.0.0.52 255.255.255.0 0.0.0.2 255.255.255.0 (Flows: N/A, Packets: N/A)
deny ip any any (Flows: N/A, Packets: N/A)
```

Applying an ACL to a virtual interface in a protocol- or subnet-based VLAN

By default, when you apply an ACL to a virtual interface in a protocol-based or subnet-based VLAN, the ACL takes effect on all protocol or subnet VLANs to which the untagged port belongs. To prevent the Ruckus device from denying packets on other virtual interfaces that do not have an ACL applied, configure an ACL that permits packets in the IP subnet of the virtual interface in all protocol-based or subnet-based VLANs to which the untagged port belongs. The following is an example configuration.

```
device# configure terminal
device(config)# vlan 1 name DEFAULT-VLAN by port
device(config-vlan-1)# ip-subnet 192.168.10.0 255.255.255.0
device(config-vlan-ip-subnet)# static ethernet 1
```

```
device(config-vlan-ip-subnet)# router-interface ve 10
device(config-vlan-ip-subnet)# ip-subnet 10.15.1.0 255.255.255.0
device(config-vlan-ip-subnet)# static ethernet 1
device(config-vlan-ip-subnet)# router-interface ve 20
device(config-vlan-ip-subnet)# logging console
device(config-vlan-ip-subnet)# exit
device(config-vlan-1)# no vlan-dynamic-discovery
Vlan dynamic discovery is disabled
device(config-vlan-1)# interface ethernet 2
device(config-if-e1000-2)# disable
device(config-if-e1000-2)# interface ve 10
device(config-vif-10)# ip address 192.168.10.254 255.255.255.0
device(config-vif-10)# interface ve 20
device(config-vif-20)# ip access-group test1 in
device(config-vif-20)# ip address 10.15.1.10 255.255.255.0
device(config-vif-20)# exit
device(config)# ip access-list extended test1
device(config-ext-nacl)# permit ip 10.15.1.0 0.0.0.255 any log
device(config-ext-nacl)# permit ip 192.168.10.0 0.0.0.255 any log
device(config-ext-nacl)# end
device#
```

Enabling strict control of ACL filtering of fragmented packets

For strict control of ACL filtering of fragmented packets, you can configure the port to drop all packet fragments. To do so, enter commands such as the following.

```
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# ip access-group frag deny
```

This option begins dropping all fragments received by the port as soon as you enter the command. This option is especially useful if the port is receiving an unusually high rate of fragments, which can indicate a hacker attack.

ACL filtering by VLAN or VE port membership

You can apply an inbound IPv4 ACL to specific VLAN members on a port (Layer 2 devices only) or to specific ports on a virtual interface (VE) (Layer 3 Devices only).

NOTE
Such specific ACL application is supported only for IPv4 ACLs, and only for inbound traffic.

By default, this feature support is disabled. To enable it, enter the following commands in global configuration mode.

```
device(config)# enable ACL-per-port-per-vlan
device(config)# write memory
device(config)# exit
device# reload
```

Configuration notes for per-port per-VLAN ACLs

- Ruckus ICX devices do not support a globally configured PBR policy together with per-port-per-VLAN ACLs.
- IPv4 ACLs that filter based on VLAN membership or VE port membership (per-port-per-VLAN), are supported together with IPv6 ACLs on the same device, as long as they are not bound to the same port or virtual interface.
- (Router image only) You cannot change VLAN membership on a port while **per-port-per-vlan** is enabled.

Applying an IPv4 ACL to VLAN members on a port (Layer 2 only)

When you bind an IPv4 ACL to a port, the port filters all inbound traffic on the port. However, on a tagged port, there may be a need to treat packets for one VLAN differently from packets for another VLAN. In this case, you can configure a tagged port on a Layer 2 device to filter packets based on the VLAN membership of the packets.

To apply an IPv4 ACL to a specific VLAN on a port, enter commands such as the following.

```
device(config)# enable ACL-per-port-per-vlan
...
device(config)# vlan 12 name vlan12
device(config-vlan-12)# untagged ethernet 1/1/5 to 1/1/8
device(config-vlan-12)# tagged ethernet 1/1/23 to 1/1/24
device(config-vlan-12)# exit
device(config)# ip access-list standard 10
device(config-std-nacl)# deny host 10.157.22.26 log
device(config-std-nacl)# deny 10.157.29.12 log
device(config-std-nacl)# deny host IPHost1 log
device(config-std-nacl)# permit any
device(config-std-nacl)# exit
device(config)# interface ethernet 1/1/23
device(config-if-e1000-1/1/23)# per-vlan 12
device(config-if-e1000-1/1/23-vlan-12)# ip access-group 10 in
```

NOTE
The **enable ACL-per-port-per-vlan** command must be followed by the **write memory** and **reload** commands to place the change into effect.

The commands in the example configure port-based VLAN 12, and add Ethernet ports 5 through 8 as untagged ports and Ethernet ports 23 and 24 as tagged ports to the VLAN. The commands following the VLAN configuration commands configure ACL 10. Finally, the last three commands apply ACL 10 on VLAN 12 for which Ethernet port 23 is a member.

Applying an IPv4 ACL to ports on a virtual interface (Layer 3 only)

You can specify a subset of ports within the VLAN containing a specified virtual interface when assigning an ACL to that virtual interface.

Use this specified ACL application when you do not want the IPv4 ACLs to apply to all the ports in the virtual interface VLAN or when you want to streamline IPv4 ACL performance for the VLAN.

To apply an ACL to a subset of ports within a virtual interface, enter commands such as the following.

```
device(config)# enable ACL-per-port-per-vlan
...
device(config)# vlan 10 name IP-subnet-vlan
device(config-vlan-10)# untag ethernet 1/1/1 to 1/2/12
device(config-vlan-10)# router-interface ve 1
device(config-vlan-10)# exit
device(config)# ip access-list standard 1
device(config-std-nacl)# deny host 10.157.22.26 log
device(config-std-nacl)# deny 10.157.29.12 log
device(config-std-nacl)# deny host IPHost1 log
device(config-std-nacl)# permit any
device(config-std-nacl)# exit
device(config)# interface ve 1/1/1
device(config-vif-1/1/1)# ip access-group 1 in ethernet 1/1/1 ethernet 1/1/3 ethernet 1/2/1 to 1/2/4
```

NOTE
The **enable ACL-per-port-per-vlan** command must be followed by the **write memory** and **reload** commands to place the change into effect.

The commands in the example configure port-based VLAN 10, add ports 1/1/1 through 1/2/12 to the VLAN, and add virtual routing interface 1 to the VLAN. The commands following the VLAN configuration commands configure ACL 1. Finally, the last two commands apply ACL 1 to a subset of the ports associated with virtual interface 1.

Filtering on IP precedence and ToS values

The following example configures an extended IP ACL that matches based on IP precedence.

```
device(config)# ip access-list extended 103
device(config-ext-nacl)# deny tcp 10.157.21.0/24 10.157.22.0/24 precedence internet
device(config-ext-nacl)# deny tcp 10.157.21.0/24 eq ftp 10.157.22.0/24 precedence 6
device(config-ext-nacl)# permit ip any any
```

The first entry in this ACL denies TCP traffic from the 10.157.21.x network to the 10.157.22.x network, if the traffic has the IP precedence option "internet" (equivalent to "6").

The second entry denies all FTP traffic from the 10.157.21.x network to the 10.157.22.x network, if the traffic has the IP precedence value "6" (equivalent to "internet").

The third entry permits all packets that are not explicitly denied by the other entries. Without this entry, the ACL would deny all incoming or outgoing IP traffic on the ports to which you assign the ACL.

The following example configures an IP ACL that matches based on ToS.

```
device(config)# ip access-list extended 104
device(config-ext-nacl)# deny tcp 10.157.21.0/24 10.157.22.0/24 tos normal
device(config-ext-nacl)# deny tcp 10.157.21.0/24 eq ftp 10.157.22.0/24 tos 13
device(config-ext-nacl)# permit ip any any
```

The first entry in this IP ACL denies TCP traffic from the 10.157.21.x network to the 10.157.22.x network, if the traffic has the IP ToS option "normal" (equivalent to "0").

The second entry denies all FTP traffic from the 10.157.21.x network to the 10.157.22.x network, if the traffic has the IP ToS value "13" (equivalent to "max-throughput", "min-delay", and "min-monetary-cost").

The third entry permits all packets that are not explicitly denied by the other entries. Without this entry, the ACL would deny all incoming or outgoing IP traffic on the ports to which you assign the ACL.

ACLs to filter ARP packets

NOTE
Using ACLs to filter ARP packets is not applicable to outbound trafic.

You can use ACLs to filter ARP packets. Without this feature, ACLs cannot be used to permit or deny incoming ARP packets. Although an ARP packet contains an IP address just as an IP packet does, an ARP packet is not an IP packet; therefore, it is not subject to normal filtering provided by ACLs.

When a Ruckus device receives an ARP request, the source MAC and IP addresses are stored in the device ARP table. A new record in the ARP table overwrites existing records that contain the same IP address. This behavior can cause a condition called "ARP hijacking", when two hosts with the same IP address try to send an ARP request to the Ruckus device.

Normally ARP hijacking is not a problem because IP assignments are done dynamically; however, in some cases, ARP hijacking can occur, such as when a configuration allows a router interface to share the IP address of another router interface. Because multiple VLANs and the router interfaces that are associated with each of the VLANs share the same IP segment, it is possible for two hosts in two different VLANs to fight for the same IP address in that segment. ARP filtering using ACLs protects an IP host record in the ARP table from being overwritten by a hijacking host. Using ACLs to filter ARP requests checks the source IP address

in the received ARP packet. Only packets with the permitted IP address will be allowed to be written in the ARP table; others are dropped.

Configuration considerations for filtering ARP packets

- Filtering ARP packets is available on devices running Layer 3 code. This filtering occurs on the management processor.
- Filtering ARP packets is available on physical interfaces and virtual routing interfaces. It is supported on the following physical interface types: Ethernet and trunks.
- ACLs used to filter ARP packets on a virtual routing interface can be inherited from a previous interface if the virtual routing interface is defined as a follower virtual routing interface.
- Only extended numbered ACLs with protocol IP can be used. If any other ACL is used, an error is displayed.

Configuring ACLs for ARP filtering

To configure ACLs for ARP filtering, enter commands such as the following.

```
device(config)# ip access-list extended 101
device(config-ext-nacl)# permit ip host 192.168.2.2 any
device(config-ext-nacl)# exit
device(config)# ip access-list extended 102
device(config-ext-nacl)# permit ip host 192.168.2.3 any
device(config-ext-nacl)# exit
device(config)# ip access-list extended 103
device(config-ext-nacl)# permit ip host 192.168.2.4 any
device(config-ext-nacl)# exit
device(config)# vlan 2
device(config-vlan-2)# tagged ethernet 1/1/1 to 1/1/2
device(config-vlan-2)# router-interface ve 2
device(config-vlan-2)# vlan 3
device(config-vlan-3)# tagged ethernet 1/1/1 to 1/1/2
device(config-vlan-3)#router-interface ve 3
device(config-vlan-3)# vlan 4
device(config-vlan-4)# tagged ethernet 1/1/1 to 1/1/2
device(config-vlan-4)# router-interface ve 4
device(config-vlan-4)# interface ve 2
device(config-ve-2)# ip access-group 101 in
device(config-ve-2)# ip address 192.168.2.1/24
device(config-ve-2)# ip use-ACL-on-arp 103
device(config-ve-2)# exit
device(config)# interface ve 3
device(config-ve-3)# ip access-group 102 in
device(config-ve-3)# ip follow ve 2
device(config-ve-3)# ip use-ACL-on-arp
device(config-ve-3)# exit
device(config-vlan-4)# interface ve 4
device(config-ve-4)# ip follow ve 2
device(config-ve-4)# ip use-ACL-on-arp
device(config-ve-4)# exit
```

When the **ip use-acl-on-arp** command is configured, the ARP module checks the source IP address of the ARP request packets received on the interface. It then applies the specified ACL policies to the packet. Only the packet with the IP address that the ACL permits will be allowed to be written in the ARP table; those that are not permitted will be dropped.

The ACL number identifies the ID of the standard ACL that will be used to filter the packet. Only the source and destination IP addresses will be used to filter the ARP packet. The ACL number can be handled in one of two ways:

- Enter an ACL ID to explicitly specify the ACL to be used for filtering. In the example, the **ip use-ACL-on-arp 103** command specifies ACL 103 to be used as the filter.
- Allow the ACL ID to be inherited from the IP ACLs that have been defined for the device. In the example, the **ip use-ACL-on-arp** command allows the ACL to be inherited from IP ACL 101 because of the **ip follow** relationship between virtual

routing interface 2 and virtual routing interface 4. Virtual routing interface 2 is configured with IP ACL 101; thus virtual routing interface 4 inherits IP ACL 101.

ARP requests will not be filtered by ACLs if one of the following conditions occur:

- If the ACL is to be inherited from an IP ACL, but there is no IP ACL defined.
- An ACL ID is specified for the **ip use-ACL-on-arp** command, but no IP address or "any any" filtering criteria have been defined under the ACL ID.

Displaying ACL filters for ARP

To determine which ACLs have been configured to filter ARP requests, enter the **show acl-on-arp** command.

```
device(config)# show ACL-on-arp
Port ACL ID Filter Count
2    103    10
3    102    23
4    101    12
```

If a port is not specified, all ports on the device that use ACLs for ARP filtering will be included in the display.

The Filter Count column in the command output shows how many ARP packets have been dropped on the interface since the last time the count was cleared.

Clearing the filter count

To clear the filter count for all interfaces on the device, enter the **clear acl-on-arp** command.

```
device(config)# clear ACL-on-arp
```

The command resets the filter count to zero on all interfaces in a device.

QoS options for IP ACLs

ACL Quality of Service (QoS) parameters enable you to perform QoS for packets that match the ACLs.

Using an ACL to perform QoS is an alternative to directly setting the internal forwarding priority based on incoming port, VLAN membership, and so on. The following QoS ACL options are supported:

- **dscp-cos-mapping**: This option is similar to the **dscp-matching** command. This option maps the DSCP value in incoming packets to a hardware table that provides mapping of each of the 0 through 63 DSCP values, and distributes them among eight traffic classes (internal priorities) and eight 802.1p priorities.

  By default, the FastIron device does the 802.1p-to-CoS mapping. If you want to change the priority mapping to DSCP-to-CoS, you must enter the following ACL rule:

  ```
  permit ip any any dscp-cos-mapping
  ```

- **dscp-marking**: Marks the DSCP value in the outgoing packet with the value you specify.
- **internal-priority-marking** and **802.1p-priority-marking**: Supported with the DSCP marking option, these commands assign traffic that matches the ACL to a hardware forwarding queue (**internal-priority-marking**), and remark the packets that match the ACL with the 802.1p priority (**802.1p-priority-marking**).
- **dscp-matching**: Matches on the packet DSCP value. This option does not change the packet forwarding priority through the device or mark the packet.
- **802.1p-priority-matching**: Inspects the 802.1p bit in the ACL that can be used with adaptive rate limiting.

NOTE
For more information, refer to the *Ruckus FastIron Command Reference* "sequence (permit|deny in extended ACLs)" topic.

Configuration notes for QoS options

FastIron devices do not support marking and prioritization simultaneously with the same rule (and do not support DSCP CoS mapping at all). Instead, you must create two separate rules. In other words, you can mark a rule with DSCP or 802.1p information, or you can prioritize a rule based on DSCP or 802.1p information. You can enable only one of the following ACL options per rule:

- **802.1p-priority-marking**
- **dscp-marking**
- **internal-priority-marking**

For example, any one of the following commands is supported.

```
device(config)# ip access-list extended 101
device(config-ext-nacl)# permit ip any any dscp-marking 43
```

or

```
device(config)# ip access-list extended 101
device(config-ext-nacl)# permit ip any any internal-priority-marking 6
```

or

```
device(config)# ip access-list extended 101
device(config-ext-nacl)# permit ip any any dscp-marking 43 802.1p-priority-marking 4 internal-priority-
marking 6
```

Using a combined ACL for 802.1p marking

FastIron devices support a simple method for assigning an 802.1p priority value to packets without affecting the actual packet or the DSCP.

When you specify 802.1p priority marking values directly, internal priority marking is optional.

NOTE
Using a combined ACL for 802.1p marking is not applicable to outbound traffic.

The following example, for any IP protocol, specifies 802.1p priority marking "1".

```
device(config)# ip access-list extended 104
device(config-ext-nacl)# permit ip any any 802.1p-priority-marking 1
```

The following example, for any IP protocol, specifies 802.1p priority marking "1" and internal priority marking "5".

```
device(config)# ip access-list extended 104
device(config-ext-nacl)# permit ip any any 802.1p-priority-marking 1 internal-priority-marking 5
```

The following example, for TCP, specifies 802.1p priority marking "1".

```
device(config)# ip access-list extended 105
device(config-ext-nacl)# permit tcp any any 802.1p-priority-marking 1
```

The following example, for TCP, specifies 802.1p priority marking "1" and internal priority marking "5".

```
device(config)# ip access-list extended 105
device(config-ext-nacl)# permit tcp any any 802.1p-priority-marking 1 internal-priority-marking 5
```

The following example, for UDP, specifies 802.1p priority marking "1".

```
device(config)# ip access-list extended 105
device(config-ext-nacl)# permit udp any any 802.1p-priority-marking 1
```

The following example, for UDP, specifies 802.1p priority marking "1" and internal priority marking "5".

```
device(config)# ip access-list extended 105
device(config-ext-nacl)# permit udp any any 802.1p-priority-marking 1 internal-priority-marking 5
```

In each of the preceding pairs of examples, the first command does not specify the internal-priority value, which means it maintains a default value of 1 (equal to the 802.1p value). In the second command, the internal-priority value is configured to 5.

Configuring QoS priority for a VLAN

Use this procedure to configure QoS priority for a VLAN.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Check the target VLAN configuration.

   ```
   device(config)# show running-config vlan 100
   vlan 100 by port
    tagged ethe 1/1/3 to 1/1/5
   !
   !
   ```

   In this example, VLAN 100 is used.

3. Enter VLAN configuration mode for the target VLAN.

   ```
   device(config)# vlan 100
   ```

4. Attach a router interface to the target VLAN.

   ```
   device(config-vlan-100)# router-interface ve 100
   ```

5. Verify the VLAN configuration.

   ```
   device(config-vif-100)# show running-config vlan 100
   vlan 100 by port
    tagged ethe 1/1/3 to 1/1/5
    router-interface ve 100
   !
   ```

6. Return to global configuration mode.

   ```
   device(config-vif-100)# exit
   ```

7. Create an extended ACL.

   ```
   device(config)# ip access-list extended acl-remark
   ```

   In this example, the extended ACL is named "acl-remark".

8. Configure a rule in the ACL with the required qualifiers and the internal priority that you want.

   ```
   device(config-ext-nacl)# permit ip any any 802.1p-and-internal-marking 5
   ```

   In this example, QoS priority 5 is used.

9. Verify the ACL configuration.

```
device(config-ext-nacl)# show access-list named-acl acl-remark

Extended IP access list  acl-remark : 1 entry
permit ip any any  802.1p-and-internal-marking 5
```

10. Go to Virtual port configuration mode.

```
device(config-ext-nacl)# interface ve 100
```

11. Apply the ACL to Virtual port VLAN 100.

```
device(config-vif-100)# ip acc acl-remark in
```

QoS priority for a VLAN configuration example

```
device# configure terminal
device(config)# show running-config vlan 100
device(config)# vlan 100
device(config-vif-100)# show running-config vlan 100
device(config-vlan-100)# router-interface ve 100
device(config-vif-100)# show running-config vlan 100
device(config-vif-100)# exit
device(config)# ip access-list extended acl-remark
device(config-ext-nacl)# permit ip any any 802.1p-and-internal-marking 5
device(config-ext-nacl)# show access-list named-acl acl-remark
device(config-ext-nacl)# int ve 100
device(config-vif-100)# ip acc acl-remark in
```

DSCP matching

The **dscp-matching** option matches on the packet DSCP value. This option does not change the packet forwarding priority through the device or mark the packet.

To configure an ACL that matches on a packet with DSCP value 29, enter a command such as the following.

```
device(config)# ip access-list extended 112
device(config-ext-nacl)# permit ip 1 0.1.1.0 0.0.0.255 10.2.2.x 0.0.0.255 dscp-matching 29
```

ACL-based rate limiting

ACL-based rate limiting provides the facility to limit the rate for inbound IP traffic that matches the permit conditions in extended IP ACLs. This feature is available in the Layer 2 and Layer 3 code.

NOTE
For implementation details, refer to the "Rate Limiting and Rate Shaping" section of the *Ruckus FastIron Traffic Management Configuration Guide*.

ACL statistics

ACL statistics is a mechanism for counting the number of packets and the number of bytes per packet to which ACL filters are applied.

To see the configuration procedures for ACL statistics, refer to "Traffic Policies" in the *Ruckus FastIron Traffic Management Configuration Guide*.

NOTE
The terms *ACL statistics* and *ACL accounting* are used interchangeably in this guide and mean the same thing.

ACL accounting

ACL accounting helps to collect usage information for access control lists configured on the device.

Counters, stored in hardware, keep track of the number of times an ACL filter is used. ACL accounting provides statistics for permit rules, deny rules, and implicit rules that help in identifying usage of particular traffic. ACL accounting is supported on IPv4 ACLs, IPv6 ACLs, and Layer 2 MAC filters and provides accounting information for inbound ACLs.

NOTE
IPv4 and IPv6 ACL accounting is supported in Policy-based Routing (PBR).

Feature limitations for ACL accounting

- Traffic Policer and ACL accounting cannot coexist.
- ACL accounting is not supported on dynamic ACLs.
- ACL accounting is not supported on management interfaces.
- ACL accounting is not supported with IP Source Guard (IPSG).

Configuring IPv4 ACL accounting

If you enable accounting for an ACL, it applies to all rules in that ACL, including implicit rules. You can enable ACL accounting for named and numbered ACLs.

1. To enable ACL accounting for a configured ACL, use the **enable-accounting** command in IPv4 ACL configuration mode.

```
device(config)# ip access-list standard 10
device(config-std-nacl)# enable-accounting
```

2. To display ACL accounting information, use the **show access-list accounting** command. The accounting statistics are collected every five seconds and synchronized to remote units once per minute.

```
device# show access-list accounting ve 16 in
IPV4 ACL Accounting Information
devNum[0] => ACL: 10
    0: permit any
     Hit Count:   (1Min)            0   (5Sec)     0
                  (PktCnt)          0 (ByteCnt)    0
    ------------------------------------------------
    65535: Implicit Rule deny any any
     Hit Count:   (1Min)            0   (5Sec)     0
                  (PktCnt)          0 (ByteCnt)    0
    ------------------------------------------------

IPV6 ACL Accounting Information
devNum[0] => ACL: v6
    0:  permit ipv6 any any
     Hit Count:   (1Min)             0   (5Sec)    0
                  (PktCnt)           0  (ByteCnt)  0
    ------------------------------------------------
    65533: Implicit ND_NA Rule: permit any any
     Hit Count:   (1Min)             0   (5Sec)    0
                  (PktCnt)           0 (ByteCnt)   0
    ------------------------------------------------
    65534: Implicit ND_NS Rule: permit any any
     Hit Count:   (1Min)             0   (5Sec)    0
                  (PktCnt)           0 (ByteCnt)   0
    ------------------------------------------------
    65535: Implicit Rule: deny any any
     Hit Count:   (1Min)             0   (5Sec)    0
                  (PktCnt)           0 (ByteCnt)   0
    ------------------------------------------------
```

3. To clear ACL accounting statistics for configured ACLs, choose one of the following options:
   - For ACLs configured on a specific interface, use the **clear access-list accounting** command in global configuration mode.
   - For all ACLs configured in the device, use the **clear access-list accounting all** command in global configuration mode.

```
device(config)# clear access-list accounting ethernet 1/1/5 in

device(config)# clear access-list accounting all
```

The following example shows how to enable ACL accounting for a numbered ACL.

```
device(config)# ip access-list standard 10
device(config-std-nacl)# permit host 10.10.10.1
device(config-std-nacl)# enable-accounting
device(config-std-nacl)# exit
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# ip access-group 10 in
```

The following example shows how to enable ACL accounting for an IPv4 named ACL.

```
device(config)# ip access-list standard myacl
device(config-std-nacl)# permit 10.10.10.0/24
device(config-std-nacl)# deny 20.20.20.0/24
device(config-std-nacl)# enable-accounting
device(config-std-nacl)# exit
device(config-std-nacl)# interface ve 121
device(config-vif-121)# ip access-group std in
```

ACLs to control multicast features

You can use ACLs to control the following multicast features:

- Limit the number of multicast groups that are covered by a static rendezvous point (RP)
- Control which multicast groups for which candidate RPs send advertisement messages to bootstrap routers
- Identify which multicast group packets will be forwarded or blocked on an interface

Enabling and viewing ACL hardware usage statistics

The number of configured ACL rules can affect the rate at which hardware resources are used.

You can use the **show access-list hw-usage on** command to enable hardware usage statistics, followed by the **show access-list** command to determine the hardware usage for an ACL. You can then modify ACL rules to better utilize hardware resources.

To enable and view hardware usage statistics, enter commands such as the following:

```
device# show access-list hw-usage on
device# show access-list 100
Extended IP access list 100 (hw usage : 2)
deny ip any any (hw usage : 1
```

The first command enables hardware usage statistics, and the second command displays the hardware usage for IP access list 100.

By default, hardware usage statistics are disabled. To disable hardware usage statistics after it has been enabled, use the **show access-list hw-usage off** command.

Displaying ACL information

For details of commands for display of IPv4 ACLs, refer to the following commands in the *Ruckus FastIron Command Reference*:

- **show access-list**
- **show access-list accounting**
- **show access-list named-acl**
- **show ip access-lists**

To display the number of Layer 4 CAM entries used by each ACL, enter the following command.

```
device# show access-list all

Extended IP access list 100 (Total flows: N/A, Total packets: N/A, Total rule cam use: 3)
permit udp host 192.168.2.169 any (Flows: N/A, Packets: N/A, Rule cam use: 1)
permit icmp any any (Flows: N/A, Packets: N/A, Rule cam use: 1)
deny ip any any (Flows: N/A, Packets: N/A, Rule cam use: 1)
```

The Rule cam use field lists the number of CAM entries used by the ACL or entry. The number of CAM entries listed for the ACL itself is the total of the CAM entries used by the ACL entries.

For flow-based ACLs, the Total flows and Flows fields list the number of Layer 4 session table flows in use for the ACL.

The Total packets and Packets fields apply only to flow-based ACLs.

Troubleshooting ACLs

Use the following methods to troubleshoot access control lists (ACLs):

- To display the number of Layer 4 CAM entries being used by each ACL, enter the **show access-list** *ACL-num* | *ACL-name* | **all** command. Refer to Displaying ACL information on page 123.
- To determine whether the issue is specific to fragmentation, remove the Layer 4 information (TCP or UDP application ports) from the ACL, and then reapply the ACL.

If you are using another feature that requires ACLs, either use the same ACL entries for filtering and for the other feature, or change to flow-based ACLs.

IPv6 ACLs

FastIron devices support IPv6 Access Control Lists (ACLs) for inbound and outbound traffic filtering.

For information on IPV6 ACL limits, refer to ACL and rule limits on page 104.

IPv6 ACL traffic filtering criteria

The Ruckus implementation of IPv6 ACLs enables traffic filtering based on the following information:

- IPv6 protocol
- Source IPv6 address
- Destination IPv6 address
- IPv6 message type
- Source TCP or UDP port (if the IPv6 protocol is TCP or UDP)
- Destination TCP or UDP port (if the IPv6 protocol is TCP or UDP)

NOTE
When setting the ACL rule to filter specific ICMP packets, the IPv6 ACL mirroring option is not supported. As a result, the **permit icmp any any echo-request mirror** command cannot be used.

IPv6 protocol names and numbers

The IPv6 protocol can be one of the following well-known names or any IPv6 protocol number from 0 through 255:

- Authentication Header (AH) protocol
- Encapsulating Security Payload (ESP)
- Internet Control Message Protocol (ICMP)
- Internet Protocol Version 6 (IPv6)
- Stream Control Transmission Protocol (SCTP)
- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

NOTE
TCP and UDP filters will be matched only if they are listed as the first option in the extension header.

For TCP and UDP, you also can specify a comparison operator and port name or number. For example, you can configure a policy to block web access to a specific website by denying all TCP port 80 (HTTP) packets from a specified source IPv6 address to the website IPv6 address.

IPv6 ACLs also provide support for filtering packets based on DSCP.

Default and implicit IPv6 ACL action

There are default actions and implicit IPv6 ACL rules.

The default action when no ACLs are configured on an interface is to permit all traffic. However, once you configure an ACL and apply it to an interface, the default action for that interface is to deny all traffic that is not explicitly permitted on the interface.

- If you want to tightly control access, configure ACLs consisting of permit rules for the access you want to permit. The ACLs implicitly deny all other access.
- If you want to secure access in environments with many users, you may want to configure ACLs that consist of explicit deny rules, with a **permit ipv6 any any** rule at the end of each ACL.

IPv6 ACLs have the following concluding implicit rules:

- **permit icmp any any nd-na**: Allows ICMP neighbor discovery acknowledgements.
- **permit icmp any any nd-ns**: Allows ICMP neighbor discovery solicitations.
- **deny ipv6 any any**: Denies IPv6 traffic. You must enter **permit ipv6 any any** as the last statement in the access list if you want to permit IPv6 traffic that was not denied by the previous statements.

  NOTE
  In an IPv6 ACL, if you do not override the implicit **deny ipv6 any any** rule, make sure to include rules that permit the IPv6 link-local address and the global unicast address. Otherwise, routing protocols such as OSPF will not work. To view the link-local address, use the **show ipv6 interface** command.

For example, if you want to deny ICMP neighbor discovery acknowledgment, then permit any remaining IPv6 traffic, enter commands such as the following.

```
device(config)# ipv6 access-list netw
device(config-ipv6-access-list netw)# permit icmp 2001:DB8:e0bb::/64
2001:DB8::/64
device(config-ipv6-access-list netw)# deny icmp any any nd-na
device(config-ipv6-access-list netw)# permit ipv6 any any
```

The first permit statement permits ICMP traffic from hosts in the 2001:DB8:e0bb::x network to hosts in the 2001:DB8::x network.

The deny statement denies ICMP neighbor discovery acknowledgement.

The last rule permits all packets that are not explicitly denied by the other rules. Without this rule, the ACL will deny all incoming IPv6 traffic on the ports to which you assigned the ACL.

Furthermore, if you add the statement **deny icmp any any** in the access list, then all neighbor discovery messages will be denied. You must explicitly enter the **permit icmp any any nd-na** and **permit icmp any any nd-ns** statements just before the **deny icmp** statement if you want the ACLs to permit neighbor discovery, as in the following example.

```
device(config)# ipv6 access-list netw
device(config-ipv6-access-list netw)# permit icmp 2001:DB8:e0bb::/64
2001:DB8::/64
device(config-ipv6-access-list netw)# permit icmp any any nd-na
device(config-ipv6-access-list netw)# permit icmp any any nd-ns
device(config-ipv6-access-list netw)# deny icmp any any
device(config-ipv6-access-list netw)# permit ipv6 any any
```

NOTE
To configure protection against spurious ND packets employed in denial of service (DoS) attacks, refer to Neighbor discovery (ND)-packet DoS attacks on page 128.

IPv6 ACL configuration notes

Consider the following configuration guidelines for IPv6 ACLs:

- IPv4 source guard and IPv6 ACLs are supported together on the same device, as long as they are not configured on the same port or virtual Interface.
- IPv6 ACLs do not support ACL filtering based on VLAN membership or VE port membership.
- IPv6 ACLs cannot be used with GRE.
- IPv6 ACLs cannot be employed to implement a user-based ACL scheme.
- If an IPv6 ACL has the implicit **deny** condition, make sure it also permits the IPv6 link-local address, in addition to the global unicast address. Otherwise, routing protocols such as OSPF will not work. To view the link-local address, use the **show ipv6 interface** command.
- IPv6 must be enabled on the interface or an IPv6 address must be configured on the interface before an ACL can be applied to it.
- (Router image only) If an ACL is applied on a VLAN physical port, if you need to change VLAN membership, you first need to remove the ACL.
- On Ruckus ICX 7850 devices, configuration of egress ACLs is blocked on any virtual interface with an associated VLAN containing an untagged port.
- On interfaces that have IPv6 ACLs applied on outbound packets, the following features are not supported:
  - ACL mirroring
  - Ruckus ICX 7150, Ruckus ICX 7650, and Ruckus ICX 7750 ACL logging
  - Traffic policies
  - Internal priority marking
- It is not possible to configure conflicting ACL filters. When configuring an ACL filter, if the sequence number already exists, or the sequence number is not specified explicitly and all the filter parameters match with an existing filter but the action does not match, the following error appears: *Error: ACL operation failed for ACL ipv6-test1 since following conflicting filter entry already exists. Please use explicit sequence number to override this error.*

Creating an IPv6 ACL

Before an IPv6 ACL can be applied to an interface, it must first be created, and IPv6 must be enabled on that interface.

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

```
device# configure terminal
```

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

```
device(config)# ipv6 access-list ipv6_test
```

3. For each rule, enter the [**sequence** *seq-num*] { **deny** | **permit** } command, specifying the needed parameters.

```
device(config-ipv6-access-list ipv6_test)# deny tcp host 2001:DB8:e0bb::2 any eq telnet
device(config-ipv6-access-list ipv6_test)# permit ipv6 any any
```

4. Access an interface on which you need to apply the ACL.

```
device(config-ipv6-access-list ipv6_test)# exit
device(config)# interface ethernet 1/1/1
```

5. If needed, enable IPv6 on that interface.

```
device(config-if-e1000-1/1/1)# ipv6 enable
```

6. Apply the ACL to the interface.

```
device(config-if-e1000-1/1/1)# ipv6 traffic-filter ipv6_test in
```

The following example creates an IPv6 ACL named "netw", with remarks preceding each rule.

```
device# configure terminal
device(config)# ipv6 access-list netw

remark Permits ICMP traffic from 2001:DB8:e0bb::x to 2001:DB8::x.
device(config-ipv6-access-list netw)# permit icmp 2001:DB8:e0bb::/64 2001:DB8::/64

remark Denies traffic from 2001:DB8:e0ac::2 to 2001:DB8:e0aa:0::24.
device(config-ipv6-access-list netw)# deny ipv6 host 2001:DB8:e0ac::2 host 2001:DB8:e0aa:0::24

remark Denies all UDP traffic.
device(config-ipv6-access-list netw)# deny udp any any

remark Permits traffic not explicitly denied by the previous rules.
device(config-ipv6-access-list netw)# permit ipv6 any any
```

The following example applies "netw" to incoming traffic on 1/1/2 and 1/4/3.

```
device# configure terminal
device(config)# interface ethernet 1/1/2
device(config-if-e1000-1/1/2)# ipv6 enable
device(config-if-e1000-1/1/2)# ipv6 traffic-filter netw in
device(config-if-e1000-1/1/2)# exit
device(config)# interface ethernet 1/4/3
device(config-if-e1000-1/4/3)# ipv6 enable
device(config-if-e1000-1/4/3)# ipv6 traffic-filter netw in
```

The following example creates an IPv6 ACL named "rtr", with remarks preceding each rule.

```
device# configure terminal
device(config)# ipv6 access-list rtr

device(config-ipv6-access-list rtr)# remark Denies TCP traffic from 2001:DB8:21::x to 2001:DB8:22::x.
device(config-ipv6-access-list rtr)# deny tcp 2001:DB8:21::/24 2001:DB8:22::/24

device(config-ipv6-access-list rtr)# remark Denies UDP traffic from UDP ports 5 through 6 to
2001:DB8:22::/24.
device(config-ipv6-access-list rtr)# deny udp any range 5 6 2001:DB8:22::/24

device(config-ipv6-access-list rtr)# remark Permits traffic not explicitly denied by the previous rules.
device(config-ipv6-access-list rtr)# permit ipv6 any any
```

The following example applies "rtr" to incoming traffic on ports 1/2/1 and 1/2/2.

```
device# configure terminal
device(config)# interface ethernet 1/2/1
device(config-if-e1000-1/2/1)# ipv6 enable
device(config-if-e1000-1/2/1)# ipv6 traffic-filter rtr in
device(config-if-e1000-1/2/1)# exit
device(config)# int eth 1/2/2
device(config-if-e1000-1/2/2)# ipv6 enable
device(config-if-e1000-1/2/2)# ipv6 traffic-filter rtr in
```

The following examples are show commands for the "rtr" ACL.

```
device# show running-config
ipv6 access-list rtr
deny tcp 2001:DB8:21::/24 2001:DB8:22::/24
deny udp any range rje 6 2001:DB8:22::/24
permit ipv6 any anyy

device# show ipv6 access-list rtr
ipv6 access-list rtr: 3 entries
10: deny tcp 2001:DB8:21::/24 2001:DB8:22::/24
20: deny udp any range rje 6 2001:DB8:22::/24
30: permit ipv6 any any
```

Enabling IPv6 on an interface to which an ACL will be applied

To enable IPv6 on an interface, enter **ipv6 enable** at the Interface level of the CLI, or assign an IPv6 address to the interface, as described in "IPv6 configuration on each router interface" in the *Ruckus FastIron Layer 3 Routing Configuration Guide*.

The following commands enable IPv6 on Ethernet interface 1/1/1 to be ready for an IPv6 ACL to be applied.

```
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# ipv6 enable
```

Applying an IPv6 ACL to a trunk group

When applying an IPv6 ACL to a trunk group, apply it to the LAG virtual interface of the trunk.

When an IPv6 ACL is applied to a LAG virtual interface in a trunk, it filters the traffic on all the member ports of the trunk.

IPv6 ACLs on virtual interfaces in protocol- or subnet-based VLANs

As with IPv4 ACLs, by default, when you apply an IPv6 ACL to a virtual interface in a protocol-based or subnet-based VLAN, the ACL takes effect on all protocol or subnet VLANs to which the untagged port belongs.

To prevent a FastIron device from denying packets on other virtual interfaces that do not have an ACL applied, configure an ACL that permits packets in the IP subnet of the virtual interface in all protocol-based or subnet-based VLANs to which the untagged port belongs.

Neighbor discovery (ND)-packet DoS attacks

Checking for ND packets with hop limit less than 255 helps protect against denial of service (DoS) attacks.

Spurious neighbor discovery (ND) packets are among the tactics employed in denial of service (DoS) attacks. Based on the assumption that the only valid packet hop-limit value is 255, you can configure IPv6 ACLs to drop such spurious ND packets. The following types of ND packets are checked:

- neighbor advertisement (NA)
- neighbor solicitation (NS)
- router advertisement (RA)
- router solicitation (RS)

IPv6 ACLs have the following concluding implicit rules, which impact on hop-limit check:

- **permit icmp any any nd-na**: Allows ICMP neighbor discovery acknowledgements.
- **permit icmp any any nd-ns**: Allows ICMP neighbor discovery solicitations.

Hop-limit check enablement varies with the type of ND packet, as indicated in the following table:

TABLE 14 Effect of ND hop-limiting on ND rules

Packet type	Implicit or configured rules	Hop-limit check
ND acknowledgement	implicit **permit**	Yes
ND acknowledgement	**permit icmp ... nd-na**	Yes
ND solicitation	implicit **permit**	Yes
ND solicitation	**permit icmp ... nd-ns**	Yes
router advertisement	**permit icmp ... router-advertisement**	Yes
router advertisement	If not specifically permitted, denied by the implicit **deny ipv6 any any**.	No
router solicitation	**permit icmp ... router-solicitation**	Yes
router solicitation	If not specifically permitted, denied by the implicit **deny ipv6 any any**.	No

Hop-limit check is not applicable to any other types of rules. For example, even if hop-limit check is enabled for the ACL, it does not apply to the two following rules:

```
device(config-ipv6-access-list nd_acl)# permit icmp any any
device(config-ipv6-access-list nd_acl)# permit ipv6 any any
```

Configuring ND-packet hop-limit check

Enabling neighbor discovery (ND)-packet hop-limit check helps protect against denial of service (DoS) attacks.

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

   ```
   device# configure terminal
   ```

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

   ```
   device(config)# ipv6 access-list nd_acl
   ```

3. If needed, add or modify ACL rules.

   ```
   device(config-ipv6-access-list nd_acl)# 10 permit udp any any
   device(config-ipv6-access-list nd_acl)# 20 permit tcp host 1000::1 host 1000::2
   device(config-ipv6-access-list nd_acl)# 30 permit icmp any any router-advertisement
   device(config-ipv6-access-list nd_acl)#40 deny ipv6 any any
   ```

4. Enter the **enable nd hop-limit** command to enable hop-limit check for the ACL.

   ```
   device(config-ipv6-access-list nd_acl)# enable nd hop-limit
   ```

   To disable hop-limit check for the ACL, enter `no enable nd hop-limit`.

5. If the ACL is not applied to the relevant interface, perform the following steps:
   a) Access the interface.

```
device(config-ipv6-access-list nd_acl)# exit
device(config)# interface ethernet 1/1/1
```

   b) If needed, enable IPv6 on that interface.

```
device(config-if-e1000-1/1/1)# ipv6 enable
```

   c) Apply the ACL to the interface.

```
device(config-if-e1000-1/1/1)# ipv6 traffic-filter nd_acl in
```

Adding a comment to an IPv6 ACL entry

You can optionally add a comment to describe entries in an IPv6 ACL. The comment appears in the output of **show** commands that display ACL information.

You can add a comment by entering the **remark** command immediately preceding an ACL entry. For example, to enter comments preceding an ACL entry, enter commands such as the following.

```
device(config)# ipv6 access-list rtr
device(config-ipv6-access-list rtr)# remark This entry permits ipv6 packets from 2001:DB8::2 to any
destination
device(config-ipv6-access-list rtr)# permit ipv6 host 2001:DB8::2 any
device(config-ipv6-access-list rtr)# remark This entry denies udp packets from any source to any destination
device(config-ipv6-access-list rtr)# deny udp any any
device(config-ipv6-access-list rtr)# remark This entry denies IPv6 packets from any source to any
destination
device(config-ipv6-access-list rtr)# deny ipv6 any any
device(config-ipv6-access-list rtr)# write memory
```

The following shows the comment text for the ACL named "rtr" in a **show running-config** display.

```
device# show running-config
ipv6 access-list rtr
 remark This entry permits ipv6 packets from 2001:DB8::2 to any destination
 permit ipv6 host 2001:DB8::2 any
 remark This entry denies udp packets from any source to any destination
 deny udp any any
 remark This entry denies IPv6 packets from any source to any destination
 deny ipv6 any any
```

Deleting a comment from an IPv6 ACL entry

To delete a comment from an IPv6 ACL entry, enter commands such as the following:

```
device(config)# ipv6 access-list rtr
        device(config-ipv6-access-list rtr)# no remark This entry permits ipv6 packets
from 2001:DB8::2 to any destination
```

Configuring IPv6 ACL accounting

If you enable accounting for an ACL, it applies to all rules in that ACL, including implicit rules.

1. Enable IPv6 ACL accounting using the **enable-accounting** command.

```
device(config-ipv6-access-list v6)# enable-accounting
```

2. Display the ACL accounting information using the **show access list accounting** command.

   The accounting statistics are collected every five seconds and synchronized to remote units every one minute.

```
device# show access-list accounting ve 16 in
IPV4 ACL Accounting Information
devNum[0] => ACL: 10
    0: permit any
     Hit Count:   (1Min)            0   (5Sec)     0
                  (PktCnt)          0 (ByteCnt)    0
    ------------------------------------------------
    65535: Implicit Rule deny any any
     Hit Count:   (1Min)            0   (5Sec)     0
                  (PktCnt)          0 (ByteCnt)    0
    ------------------------------------------------

IPV6 ACL Accounting Information
devNum[0] => ACL: v6
    0:  permit ipv6 any any
     Hit Count:   (1Min)            0   (5Sec)     0
                  (PktCnt)          0  (ByteCnt)   0
    ------------------------------------------------
    65533: Implicit ND_NA Rule: permit any any
     Hit Count:   (1Min)            0   (5Sec)     0
                  (PktCnt)          0 (ByteCnt)    0
    ------------------------------------------------
    65534: Implicit ND_NS Rule: permit any any
     Hit Count:   (1Min)            0   (5Sec)     0
                  (PktCnt)          0 (ByteCnt)    0
    ------------------------------------------------
    65535: Implicit Rule: deny any any
     Hit Count:   (1Min)            0   (5Sec)     0
                  (PktCnt)          0 (ByteCnt)    0
    ------------------------------------------------
```

3. To clear ACL accounting statistics for configured ACLs, choose one of the following options:
   - For ACLs configured on a specific interface, use the **clear access-list accounting** command in global configuration mode.
   - For all ACLs configured in the device, use the **clear access-list accounting all** command in global configuration mode.

```
device(config)# clear access-list accounting ethernet 1/1/5 in

device(config)# clear access-list accounting all
```

The following example shows how to enable IPv6 ACL accounting.

```
device(config)# ipv6 access-list v6
device(config-ipv6-access-list v6)# enable-accounting
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# ipv6 enable
device(config-if-1/1/1)# ipv6 access-list v6 in
device(config)# write memory
```

Displaying IPv6 ACLs

To display the IPv6 ACLs configured on a device, enter the **show ipv6 access-list** command.

```
device# show ipv6 access-list
ipv6 access-list v6-ACL1: 1 entries
deny ipv6 any any
ipv6 access-list v6-ACL2: 1 entries
permit ipv6 any any
ipv6 access-list v6-ACL3: 2 entries
deny ipv6 2001:DB8:10::/64 any
```

```
permit ipv6 any any
ipv6 access-list v6-ACL4: 2 entries
deny ipv6 2001:DB8::/64 any
permit ipv6 any any
ipv6 access-list rate-ACL: 1 entries
permit ipv6 any any traffic-policy rate800M
ipv6 access-list v6-ACL5: 8 entries
permit tcp 2001:DB8::/64 any
permit ipv6 2001:DB8::/64 any
permit ipv6 2001:DB8:101::/64 any
permit ipv6 2001:DB8:10::/64 2001:DB8:102::/64
permit ipv6 host 2001:DB8:aa:10::102 host 2001:DB8:101::102
permit ipv6 host 2001:DB8:10::101 host 2001:DB8:101::101 dscp-matching 0
dscp-marking 63 dscp-cos-mapping
permit ipv6 any any dscp-matching 63 dscp-cos-mapping
permit ipv6 any any fragments
```

To display a specific IPv6 ACL configured on a device, enter the **show ipv6 access-list** command followed by the ACL name. The following example shows the ACL named "rtr".

```
device# show ipv6 access-list rtr
ipv6 access-list rtr: 3 entries
remark This entry permits ipv6 packets from 2001:DB8::2 to any destination
permit ipv6 host 2001:DB8::2 any
remark This entry denies udp packets from any source to any destination
deny udp any any
remark This entry denies IPv6 packets from any source to any destination
deny ipv6 any any
```

ACL logging

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

If an ACL rule matches a packet, the software generates a syslog entry and an SNMP trap; and starts a five-minute timer. The timer keeps track of all packets that match the ACL entries. After about five minutes, the software generates a syslog entry reporting the matches.

Note the following details:

- If the packet rate is high—exceeding the CPU processing rate—the packet count may be inaccurate.
- If there are no matches within the five-minute timer interval, the timer stops, restarting with the next match.
- The timer for logging packets denied by MAC address filters is a different timer than the ACL logging timer.

Configuration notes for ACL logging

Before configuring ACL logging on your device, consider the following configuration notes for IPv4 and IPv6.

- ACL logging is a CPU-intensive feature, intended for debugging purposes. Ruckus recommends that you disable ACL logging after the debug session is over.
- To maintain maximum performance, log only specific filters.
- When ACL logging is enabled, packets sent to the CPU are automatically rate-limited to prevent CPU overload.
- You can enable ACL logging on physical and virtual interfaces.
- Packets are logged in the syslog based on a sample time period.
- If you remove an egress ACL with rules enabled for logging, although new packets are not logged, previous logs will be in the syslog when the sample timer expires.
- The log count may not be accurate for broadcast traffic and for unknown unicast traffic.

- On the Ruckus ICX 7150, ICX 7650, and Ruckus ICX 7750, ACL logging is not supported for egress ACLs.
- In a rule that includes one or more of the following parameters, the **log** keyword is ignored:
  - **dscp-matching**
  - **dscp-marking**
  - **802.1p-priority-matching**
  - **802.1p-priority-marking**
  - **802.1p-and-internal-marking**
- ACL logging is not supported for dynamic ACLs with MAC authentication or 802.1X enabled.
- ACL logging is supported for ACLs that are applied to network management access features such as Telnet, SSH, and SNMP.

Configuration notes for IPv4 logging only

When an ACL that includes an entry with a logging option is applied for egress traffic to a port that has logging enabled, and then the same ACL is applied to another port on the same system, traffic on the latter port is also logged, whether logging is explicitly enabled for that latter port or not. On the other hand, when an ACL is applied to a port that has logging disabled, and then the same ACL is applied to another port on the same system, traffic on the latter port is also not logged, whether logging is explicitly enabled for that latter port or not.

Configuration tasks for ACL logging

To enable ACL logging, implement the following tasks.

1. In ACLs, create rules that contain the **log** keyword.
2. Enable ACL logging on needed interfaces.
3. Apply the ACLs to the ports on which ACL logging is enabled.

Example ACL logging configuration

The commands for ACL logging vary between IPv4 and IPv6 ACLs.

The following commands create an IPv4 ACL with rules that include the **log** keyword, enable ACL logging on an interface, and then bind the ACL to that interface to filter incoming traffic. Statistics for packets that match the rules will be logged.

```
device(config)# ip access-list standard 1
device(config-std-nacl)# deny host 10.157.22.26 log
device(config-std-nacl)# deny 10.157.29.12 log
device(config-std-nacl)# deny host IPHost1 log
device(config-std-nacl)# permit any
device(config-std-nacl)# interface ethernet 1/1/4
device(config-if-e1000-1/1/4)# acl-logging
device(config-if-e1000-1/1/4)# ip access-group 1 in
```

The following commands create an IPv6 ACL with rules that include the **log** keyword, enable ACL logging for that ACL, and then bind the ACL to a specified interface to filter incoming traffic. Statistics for packets that match the rules will be logged.

```
device(config)# ipv6 acc ACL_log_v6
device(config-ipv6-access-list ACL_log_v6)# logging-enable
device(config-ipv6-access-list ACL_log_v6)# deny ipv6 host 2001:DB8::1 any log
device(config-ipv6-access-list ACL_log_v6)# interface ethernet 1/9/12
device(config-if-e1000-1/9/12)# ipv6 traffic-filter ACL_log_v6 in
```

Displaying ACL log entries

To display syslog entries, enter the **show log** command from any CLI prompt.

```
device # show log

Syslog logging: enabled ( 0 messages dropped, 0 flushes, 0 overruns)
    Buffer logging: level ACDMEINW, 132 messages logged
    level code: A=alert C=critical D=debugging M=emergency E=error
                I=informational N=notification W=warning

Dynamic Log Buffer (4000 lines):
Mar 16 22:23:10:I:ACL: ACL: List 101 permitted 253 192.168.20.1()(VLAN: 646 Ethernet 1/1/24 0010.9400.0002)
-> 192.168.10.2(), 1 event(s)
Mar 16 22:23:10:I:ACL: ACL: List 101 permitted 253 192.168.20.3()(VLAN: 646 Ethernet 1/1/24 0010.9400.0002)
-> 192.168.10.6(), 1 event(s)
Mar 16 22:23:10:I:ACL: ACL: List 101 denied tcp 10.20.15.6()(VLAN: 646 Ethernet 1/1/24 0010.9400.0002) ->
10.20.18.6(), 1 event(s)
Mar 16 22:22:43:I:ACL 101 modified by un-authenticated user from console session
Mar 16 22:22:35:I:ACL 101 modified by un-authenticated user from console session
Mar 16 22:15:21:I:ACL: 101 applied to port VE 646 by un-authenticated user from console session
```

Sequence-based ACL editing

You can specify and modify the sequence in which ACL rules are applied to traffic.

By default, the order in which ACL rules run is determined by the sequence in which they are added to the ACL. Beginning with FastIron 08.0.50—preserving this order—sequence numbers are automatically added to existing ACL rules in the following manner:

- The first rule within each ACL is numbered 10.
- The sequence number for each succeeding rule is incremented by 10.

In new ACLs that you create, specifying rule sequence numbers is optional. However, sequence numbers are assigned automatically in the previously mentioned order.

Sequence numbers have the following advantages:

- If you need a new rule to run between existing rules, you assign the new rule a sequence number between those two rules.
- New rules are implemented seamlessly, with no need to re-apply ACLs.
- If you delete a rule, there is no need to re-apply the ACL.

If you add a rule to an ACL without specifying its sequence, the rule is added at the end of the list. Such a rule is automatically assigned the next multiple of 10 as a sequence number.

The following actions are also supported:

- Suppressing sequence numbers towards a downgrade on page 137
- Regenerating sequence numbers within a specified ACL. Although by default, such regeneration assigns 10 to the first rule and increments each succeeding rule by 10, you can specify both parameters. Sequence regeneration settings (first sequence number and sequence interval number) are persistent, even following reload of the active unit.

Inserting rules into IPv4 ACLs

To insert rules into an IPv4 ACL, complete the following steps.

The examples in this topic are for a named standard IPv4 ACL. Modify as needed for ACLs that are numbered and extended.

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

```
device# configure terminal
```

2. Enter the **ip access-list** command to access ACL configuration mode for the ACL you need to modify.

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

3. Enter the **show ip access-list** command to display the rules of the relevant ACL.

```
device(config-std-nacl)# show ip access-list ip_acl_1
Standard IP access list ip_acl_1: 3 entries
10: permit 1.1.1.0  0.0.0.255
11: permit host 2.2.2.1
20: deny   2.2.2.0  0.0.0.255
```

4. To insert a rule between two adjacent rules, do the following:
   a) Enter the **regenerate-seq-number** command.

```
device(config-std-nacl)# regenerate-seq-number
```

   b) Re-enter the **show ip access-list** command.

```
device(config-std-nacl)# show ip access-list ip_acl_1
Standard IP access list ip_acl_1: 3 entries
10: permit 1.1.1.0  0.0.0.255
20: permit host 2.2.2.1
30: deny   2.2.2.0  0.0.0.255
```

5. Create the new rule, specifying the needed sequence number.

```
device(config-std-nacl)# sequence 15 permit 2.1.1.3
```

Inserting rules into IPv6 ACLs

To insert rules into an IPv6 ACL, complete the following steps.

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

```
device# configure terminal
```

2. Enter the **ipv6 access-list** command to access ACL configuration mode for the ACL you need to modify.

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

3. Enter the **show ipv6 access-list** command to display the rules of the relevant ACL.

```
device(config-ipv6-access-list ipv6_acl_1)#  show ipv6 access-list ipv6_acl_1
ipv6 access list ipv6_acl_1: 3 entries
10: permit ipv6 2001:DB8::/64 any
11: permit ipv6 2001:DB8:101::/64 any
20: permit ipv6 host 2001:DB8:aa:10::102 host 2001:DB8:101::102
```

4. To insert a rule between two adjacent rules, do the following:
   a) Enter the **regenerate-seq-number** command.

```
device(config-ipv6-access-list ipv6_acl_1)# regenerate-seq-number
```

   b) Re-enter the **show ip access-list** command.

```
device(config-ipv6-access-list ipv6_acl_1)# show ipv6 access-list ipv6_acl_1
10: permit ipv6 2001:DB8::/64 any
20: permit ipv6 2001:DB8:101::/64 any
30: permit ipv6 host 2001:DB8:aa:10::102 host 2001:DB8:101::102
```

5. Create the new rule, specifying the needed sequence number.

```
device(config-ipv6-access-list ipv6_acl_1)# sequence 25 permit ipv6 2001:DB8:10::/64
2001:DB8:102::/64
```

Deleting IPv4 ACL rules

To delete rules from an IPv4 ACL, complete the following steps.

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

```
device# configure terminal
```

2. Enter the **ip access-list** command to access ACL configuration mode for the ACL you need to modify.

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

3. If needed, enter the **show ip access-list** command to display the rules of the relevant ACL.

```
device(config-std-nacl)# show ip access-list ip_acl_1
Standard IP access list ip_acl_1: 3 entries
10: permit 1.1.1.0  0.0.0.255
15: permit 2.1.1.3
20: permit host 2.2.2.1
30: deny   2.2.2.0  0.0.0.255
```

4. To delete a rule by sequence number, enter the **no sequence** command, specifying the sequence number.

```
device(config-std-nacl)# no sequence 15
```

5. To delete a rule by content, enter the **no permit** command, specifying the rule content.

```
device(config-std-nacl)# no permit host 2.2.2.1
```

6. If you need to re-use a sequence number that you deleted, enter the **regenerate-seq-number** command.

```
device(config-std-nacl)# regenerate-seq-number
```

Deleting IPv6 ACL rules

To delete rules from an IPv6 ACL, complete the following steps.

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

```
device# configure terminal
```

2. Enter the **ipv6 access-list** command to access ACL configuration mode for the ACL you need to modify.

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

3. If needed, enter the **show ipv6 access-list** command to display the rules of the relevant ACL.

```
device(config-ipv6-access-list ipv6_acl_1)# show ipv6 access-list ipv6_acl_1
10: permit ipv6 2001:DB8::/64 any
20: permit ipv6 2001:DB8:101::/64 any
25: permit ipv6 2001:DB8:10::/64 2001:DB8:102::/64
30: permit ipv6 host 2001:DB8:aa:10::102 host 2001:DB8:101::102
```

4. To delete a rule by sequence number, enter the **no sequence** command, specifying the sequence number.

```
device(config-ipv6-access-list ipv6_acl_1)#  no sequence 25
```

5. To delete a rule by content, precede the complete rule with **no**.

```
device(config-ipv6-access-list ipv6_acl_1)# no permit ipv6 2001:DB8:101::/64 any
```

6. If you need to re-use a sequence number that you deleted, enter the **regenerate-seq-number** command.

```
device(config-std-nacl)# regenerate-seq-number
```

Suppressing sequence numbers towards a downgrade

To downgrade to a release earlier than FastIron 8.0.50, suppress ACL rule sequence numbers by completing the following steps.

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

```
device# configure terminal
```

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

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

3. Enter the **suppress-acl-seq** command.

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

Policy-Based Routing

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 access control lists (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 its 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 Access Control Lists chapter.

The following are the basic steps used to configure a PBR policy:

1. Configure ACLs that specify all the conditions required to match the desired packets to be routed using PBR.
2. Configure a route map that matches on the ACLs and sets the route information.
3. Enable PBR by applying the route map globally or on an untagged interface or virtual interface.

PBR permit and deny actions

To configure PBR, you define the policies using ACLs and route maps, and 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 user-applied 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 15 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”

TABLE 15 Behavior of Permit and Deny actions in different contexts (continued)

Route-map level permit and deny actions	ACL clause permit and deny actions	Resulting Ternary Content Addressable Memory (TCAM) action
		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.

LAG formation with PBR policy

PBR can be applied on a LAG virtual interface only if the port is untagged. When a LAG is formed, the PBR policy on the LAG virtual interface applies to all the member ports. If a different PBR policy exists on a member port at the time of a LAG formation, that policy is overridden by the PBR policy on the LAG virtual interface. If the LAG virtual interface does not have a PBR policy, then the member ports will not have a PBR policy.

When a LAG is removed, the PBR policy that was applied to the LAG virtual interface is unbound (removed) from former member ports. If global PBR is configured, the member ports adhere to the global PBR; otherwise, no PBR policy is bound to former member ports. When a LAG is formed, all ports must have the same PBR configuration. During LAG creation, the configuration on the LAG virtual interface is replicated to all ports.

Route map

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

The IP standard or extended ACLs can be used to establish the match criteria. Using the standard ACLs with route maps in PBR, an IP packet is routed based on its source IP address. Using the 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 available 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 of preference. If no next hops are reachable, the egress interface is selected based on the IP route table. 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 route-map 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 matched traffic:

- IPv4 address
- IPv6 address
- Null interface
- VRF interface

Configuration guidelines for IPv4 PBR

- PBR is supported in the full Layer 3 code only.
- PBR is not supported together with ingress ACLs on the same port.
- Global PBR is not supported with per-port-per-VLAN ACLs.
- Global PBR is supported only on the default VRF.
- A PBR policy on an interface takes precedence over a global PBR policy.
- Although you can apply an IPv4 policy route map to more than 64 interfaces, only the first 64 active route maps (including instances) will be used.
- You cannot apply PBR on a port if that port already has ingress ACLs, ACL-based rate limiting, DSCP-based QoS, or MAC address filtering.
- The number of route maps that you can define is limited by the available system memory, which is determined by the system configuration and how much memory other features use. When a route map is used in a PBR policy, the PBR policy uses up to 21 instances of a route map, up to five ACLs in a matching policy of each route map instance, and up to six next hops in a set policy of each route map instance. The CLI allows you to configure more than 21 route maps instances; however, only 21 instances can be active at one time. An active route map is a route map with at least one valid ACL and one valid next hop. The CLI also allows you to configure more than six next hops in a route map; however, the extra next hops will not be placed in the PBR database. A route map might be used by other features such as BGP or OSPF, which may use more than six next hops.
- Denial of Service (DoS) protection is supported in PBR.
- IPv4 ACL accounting is supported in PBR.

- ACLs with the **log** option configured in the **access-list** command should not be used for PBR purposes.
- PBR ignores explicit or implicit **deny ip any any** ACL entries to ensure that the traffic is compared to all ACLs for route maps that use multiple ACLs. PBR also ignores any deny clauses in an ACL. Traffic that matches a deny clause is routed normally using Layer 3 paths.
- PBR always selects the first next hop from the next-hop list that is up. If a PBR policy next hop goes down, the policy uses another next hop if available. If no next hops are available, the device routes the traffic in the normal way.
- PBR is not supported for fragmented packets. If the PBR ACL filters on Layer 4 information such as TCP/UDP ports, fragmented packets are routed normally.
- You can change route maps or ACL definitions dynamically and do not need to rebind the PBR policy to an interface.

Configuring an IPv4 PBR policy with an 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. This task uses Access Control Lists (ACLs), which are explained in greater detail in the *Ruckus FastIron Security Configuration Guide* for your platform.

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)# access-list 99 permit 10.157.23.0 0.0.0.255
   ```

3. 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
   ```

4. Add IPv4 ACLs to match the IP address that is permitted by the ACL.

   ```
   device(config-routemap test-route)# match ip address 99
   ```

5. 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-routemap test-route)# set ip next-hop 192.168.3.1
   ```

   NOTE
   If the IP address used in this command is the IP address of a configured IPsec or GRE tunnel, the configuration will choose IPsec or GRE tunnel interface 192.168.3.1 as the next-hop address for matching packets. If you want to set the next hop using a GRE tunnel or IPsec tunnel, use the **set next-hop-ip-tunnel** command.

   Optionally, the route map can be configured to forward the packet to the neighbor router without decrementing the Time-to-Live (TTL) value in the packet header for the traffic matched by the policy using the **no-ttl-decrement** option.

   By default, the TTL value in the packet header is decremented (decreased) for routed traffic and the packet will be discarded when the TTL is exhausted. TTL functions as a hop count limit and every routing hop decrements the TTL value by one. When the TTL value becomes zero, the packet is discarded to prevent routing loops. The **no-ttl-decrement** option in the **set ip next-hop** command disables the TTL decrement and the packets will be forwarded without decrementing TTL for the traffic matched by the policy.

6. Enter the **exit** command to return to global configuration mode.

   ```
   device(config-routemap test-route)# exit
   ```

7. Enable PBR by applying the route map globally or on an untagged interface or virtual interface.
   - Enable IPv4 PBR globally to apply the route map to all interfaces.

     ```
     device(config)# ip policy route-map test-route
     ```

   - Enable IPv4 PBR locally by applying the route map on an interface.

     ```
     device(config)# interface ethernet 1/1/3
     device(config-if-e1000-1/1/3)# ip policy route-map test-route
     ```

   - Enable IPv4 PBR locally by applying the route map on a virtual interface.

     ```
     device(config)# interface ve 1
     device(config-vif-1)# ip policy route-map test-route
     ```

The following example shows the configuration steps to configure an IPv4 PBR policy by setting an IPv4 address as the next hop in the route map.

```
device# configure terminal
device(config)# access-list 99 permit 10.157.23.0 0.0.0.255
device(config)# route-map test-route permit 99
device(config-routemap test-route)# match ip address 99
device(config-routemap test-route)# set ip next-hop 192.168.3.1
device(config-routemap test-route)# exit
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ip policy route-map test-route
device(config-if-e1000-1/1/3)# end
```

The following example shows the configuration steps to configure an IPv4 PBR policy in which the route map is configured to forward the packet without decrementing the Time-to-Live (TTL) value in the packet header.

```
device(config)# access-list 99 permit 10.157.23.0 0.0.0.255
device(config)# route-map test-route permit 99
device(config-routemap test-route)# match ip address 99
device(config-routemap test-route)# set ip next-hop 192.168.3.1 no-ttl-decrement
device(config-routemap test-route)# exit
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ip policy route-map test-route
device(config-if-e1000-1/1/3)# end
```

Configuring an IPv4 PBR policy with the NULL0 interface as the next hop

The following steps configure an IPv4 PBR policy by setting the 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)# access-list 99 permit 10.157.23.0 0.0.0.255
   ```

3. 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
   ```

4. Add IPv4 ACLs to match the IP address that is permitted by the ACL.

```
device(config-routemap test-route)# match ip address 99
```

5. 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-routemap test-route)# set interface null0
```

6. Enter the **exit** command to return to global configuration mode.

```
device(config-routemap test-route)# exit
```

7. Enter configuration mode on the interface where you want to enable PBR by applying the route map.

```
device(config)# interface ethernet 1/1/3
```

   PBR can be enabled globally by which the route map is applied to all interfaces using the **ip policy route-map** command from the global configuration mode.

8. Enable PBR on the interface and specify the route map to be used.

```
device(config-if-e1000-1/1/3)# ip policy route-map test-route
```

The following example shows the configuration steps to configure an IPv4 PBR policy to send all traffic from 10.157.23.0 0.0.0.255 to the NULL0 interface, thus dropping the traffic instead of forwarding it.

```
device# configure terminal
device(config)# access-list 99 permit 10.157.23.0 0.0.0.255
device(config)# route-map test-route permit 99
device(config-routemap test-route)# match ip address 99
device(config-routemap test-route)# set set interface null0
device(config-routemap test-route)# exit
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ip policy route-map test-route
device(config-if-e1000-1/1/3)# end
```

Configuring an IPv4 PBR policy with a tunnel as the next hop

Traffic can be configured to route over an IPsec tunnel or GRE tunnel using PBR. The following steps configure an IPsec or GRE tunnel interface as the next hop of a PBR route map.

You must configure the IPsec tunnel or GRE tunnel before configuring the traffic to route over a tunnel. For more information about IPsec tunnel configuration, refer to the Routing traffic over an IPsec tunnel using PBR on page 252 task. For more information about GRE tunnel configuration, refer to the *Ruckus FastIron Layer 3 Routing Configuration Guide*.

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)# access-list 99 permit 10.157.23.0 0.0.0.255
```

3. 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
```

4. Add IPv4 ACLs to match the IP address that is permitted by the ACL.

```
device(config-routemap test-route)# match ip address 99
```

5. Set the configured tunnel as the next hop for a route map.

```
device(config-routemap test-route)# set next-hop-ip-tunnel 1
```

6. Enter the **end** command to return to global configuration mode.

```
device(config-routemap test-route)# end
```

7. Enter configuration mode on the interface where you want to enable IPv4 PBR by applying the route map.

```
device(config)# interface ethernet 1/1/3
```

   IPv4 PBR can be enabled globally by which the route map is applied to all interfaces using the **ip policy route-map** command from global configuration mode.

8. Enable PBR on the interface and specify the route map to be used.

```
device(config-if-e1000-1/1/3)# ip policy route-map test-route
```

The following example shows the configuration steps to route routing traffic over an IPsec tunnel using PBR.

```
device(config)# interface tunnel 1
device(config-tnif-1)# vrf forwarding blue
device(config-tnif-1)# tunnel source ethernet 1/1/1
device(config-tnif-1)# tunnel destination 10.2.2.1
device(config-tnif-1)# tunnel mode ipsec ipv4
device(config-tnif-1)# tunnel protection ipsec profile prof-blue
device(config-tnif-1)# ip address 10.4.4.4/24
device(config-tnif-1)# exit
device(config)# access-list 99 permit 10.157.23.0 0.0.0.255
device(config)# route-map test-route permit 99
device(config-routemap test-route)# match ip address 99
device(config-routemap test-route)# set next-hop-ip-tunnel 1
device(config-routemap test-route)# end
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# vrf forwarding blue
device(config-if-e1000-1/1/3)# ip policy route-map test-route
device(config-if-e1000-1/1/3)# end
```

The following example shows the configuration steps to route routing traffic over a GRE tunnel using PBR.

```
device(config)# interface tunnel 1
device(config-tnif-1)# vrf forwarding blue
device(config-tnif-1)# tunnel source ethernet 1/1/1
device(config-tnif-1)# tunnel destination 10.2.2.1
device(config-tnif-1)# tunnel mode gre ip
device(config-tnif-1)# ip address 10.4.4.4/24
device(config-tnif-1)# exit
device(config)# access-list 99 permit 10.157.23.0 0.0.0.255
device(config)# route-map test-route permit 99
device(config-routemap test-route)# match ip address 99
device(config-routemap test-route)# set next-hop-ip-tunnel 1
device(config-routemap test-route)# end
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# vrf forwarding blue
device(config-if-e1000-1/1/3)# ip policy route-map test-route
device(config-if-e1000-1/1/3)# end
```

Configuring an IPv4 PBR policy by setting a VRF-aware next hop in a route map

The following steps configure a next hop in a VRF for a route map.

A VRF must be configured and an interface must be assigned to the VRF before setting a VRF-aware next hop in a 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)# access-list 99 permit 10.157.23.0 0.0.0.255
```

3. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all of the match clauses are met.

```
device(config)# route-map test-route permit 99
```

4. Add IPv4 ACLs to match the IP address that is permitted by the ACL.

```
device(config-routemap test-route)# match ip address 99
```

5. Set the configured VRF as the next hop for a route map.

```
device(config-routemap test-route)# set ip next-hop 192.168.3.1 vrf vrf_c
```

Configuration Guidelines for IPv6 PBR

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.

The next hop can be configured as an IPv6 address, a tunnel ID, or as a drop interface. IPv6 ACLs can be configured as match criteria for the packets. An IPv6 policy route map can be applied globally or on an interface. If an IPv6 PBR policy is configured on an interface, it will take precedence over the global level IPv6 PBR policy configuration.

The **ip policy route-map** and the **ipv6 policy route-map** commands are independent of each other and can be applied on an interface and globally in any order, individually or simultaneously. Although IPv4 and IPv6 PBR work independently, and route maps can support IPv4 and IPv6 next hops and ACLs, the command used to apply the route map does affect which next hops and ACLs are supported:

- If a route-map is applied using the **ip policy route-map** command, only IPv4 next-hops and ACLs will be used and IPv6 next-hops and ACLs will be skipped.
- If a route-map is applied using the **ipv6 policy route-map** command, only IPv6 next-hops and ACLs will be used and IPv4 next-hops and ACLs will be skipped.

IPv6 PBR follows the same configuration considerations as IPv4 PBR with the following exceptions:

- Each IPv6 PBR route-map instance can support up to six IPv6 ACLs.
- IPv6 PBR is supported for the following criteria:
  - VRF support
  - Accounting
  - Stacking or HA
  - SPX

Configuring an IPv6 PBR policy with an IPv6 address as the next hop

The following steps configure an IPv6 address as the next hop in the route map in a policy. This task uses Access Control Lists (ACLs), which are explained in greater detail in the "ACLs" chapter.

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 acl8 permit 2001:DB8:12d:1300::/64
```

3. 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 acl8
```

4. Add IPv6 ACLs to match the IP address that is permitted by the ACL.

```
device(config-routemap test-route)# match ipv6 address acl8
```

5. 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-routemap test-route)# set ipv6 next-hop 2001:DB8:12d:1300:1
```

6. Enter the **exit** command to return to global configuration mode.

```
device(config-routemap test-route)# exit
```

7. Enable PBR by applying the route map globally or on an untagged interface or virtual interface.
   - Enable IPv6 PBR globally to apply the route map to all interfaces.

```
device(config)# ipv6 policy route-map test-route
```

   - Enable IPv6 PBR locally by applying the route map on an interface.

```
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ipv6 policy route-map test-route
```

   - Enable IPv6 PBR locally by applying the route map on a virtual interface.

```
device(config)# interface ve 1
device(config-vif-1)# ipv6 policy route-map test-route
```

The following example shows the configuration steps to configure a PBR policy by setting an IPv6 address as the next hop in the route map.

```
device# configure terminal
device(config)# ipv6 access-list acl8 permit 2001:DB8:12d:1300::/64
device(config)# route-map test-route permit acl8
device(config-routemap test-route)# match ipv6 address acl8
device(config-routemap test-route)# set ipv6 next-hop 2001:DB8:12d:1300:1
device(config-routemap test-route)# exit
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ipv6 policy route-map test-route
device(config-if-e1000-1/1/3)# end
```

Configuring an IPv6 PBR policy with the NULL0 interface as the next hop

Perform the following steps to configure an IPv6 PBR to set the NULL0 interface as the next hop in the route map.

The NULL0 interface does not forward traffic or receive traffic. Setting the next hop as the NULL0 interface drops the packet instead of forwarding the packet.

1. 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 v6ACL1
```

3. Define a permit clause with an IPv6 address.

```
device(config-ipv6-std)# seq 10 permit 2001:db8:85a3:0:0:8a2e:370:7334
```

4. Define a deny clause.

```
device(config-ipv6-std)# seq 11 deny any
```

5. Return to global configuration mode.

```
device(config-ipv6-std)# exit
```

6. 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 null-route permit v6ACL1
```

7. Add IPv6 ACLs to match the IP address that is permitted by the ACL.

```
device(config-routemap null-route)# match ipv6 address v6ACL1
```

8. Set the next hop as the NULL0 interface to send the traffic to the null interface, and this action will drop the packet instead of forwarding it.

```
device(config-routemap null-route)# set interface null0
```

9. Enter the **exit** command to return to global configuration mode.

```
device(config-routemap null-route)# exit
```

10. Enter configuration mode on the interface where you want to enable PBR by applying the route map.

```
device(config)# interface ethernet 1/1/3
```

    IPv6 PBR can be enabled globally by which the route map is applied to all interfaces using the **ip policy route-map** command from global configuration mode.

11. Enable PBR on the interface and specify the route map to be used.

```
device(config-if-e1000-1/1/3)# ipv6 policy route-map null-route
```

The following example shows the configuration steps to configure a PBR policy to send all IPv6 traffic from 2001:db8:85a3:0:0:8a2e:370:7334 to the NULL0 interface, thus dropping the traffic instead of forwarding it.

```
device# configure terminal
device(config)# ipv6 access-list standard v6ACL1
device(config-ipv6-std)# seq 10 permit 2001:db8:85a3:0:0:8a2e:370:7334
device(config-ipv6-std)# seq 11 deny any
device(config-ipv6-std)# exit
device(config)# route-map null-route permit v6ACL1
device(config-routemap null-route)# match ip address v6ACL1
device(config-routemap null-route)# set interface null0
device(config-routemap null-route)# exit
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ipv6 policy route-map null-route
device(config-if-e1000-1/1/3)# end
```

Configuring an IPv6 PBR policy with a tunnel as the next hop

Traffic can be configured to route IPv6 packets over an IPsec tunnel or GRE tunnel using PBR. The following steps configure an IPsec or GRE tunnel interface as the next hop of a PBR route map.

You must configure the IPsec tunnel or GRE tunnel before configuring the traffic to route over a tunnel. For more information about IPsec tunnel configuration, refer to the Routing traffic over an IPsec tunnel using PBR on page 252 task. For more information about GRE tunnel configuration, refer to the *Ruckus FastIron Layer 3 Routing Configuration Guide*.

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 99 permit 2001:DB8:90::22/64
   ```

3. 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-ipv6-access-list)# route-map tunnel-route permit 99
   ```

4. Add IPv6 ACLs to match the IP address that is permitted by the ACL.

   ```
   device(config-routemap tunnel-route)# match ipv6 address 99
   ```

5. Set the configured tunnel as the next hop for a route map.

   ```
   device(config-routemap tunnel-route)# set next-hop-ipv6-tunnel 1
   ```

6. Enter the **end** command to return to global configuration mode.

   ```
   device(config-routemap tunnel-route)# end
   ```

7. Enter configuration mode on the interface where you want to enable the IPv6 PBR policy by applying the route map.

   ```
   device(config)# interface ethernet 1/1/3
   ```

   The IPv6 PBR policy can be enabled globally when the route map is applied to all interfaces using the **ipv6 policy route-map** command from global configuration mode.

8. Enable PBR on the interface and specify the route map to be used.

   ```
   device(config-if-e1000-1/1/3)# ipv6 policy route-map tunnel-route
   ```

The following example shows the configuration steps to forward IPv6 routing traffic over an IPsec tunnel using PBR. The initial VRF and IPsec tunnel configuration is included to give a complete example.

```
device(config)# interface tunnel 1
device(config-tnif-1)# vrf forwarding marketing
device(config-tnif-1)# tunnel source ethernet 1/1/1
device(config-tnif-1)# tunnel destination 2001:DB8:2::1
device(config-tnif-1)# tunnel mode ipsec ipv6
device(config-tnif-1)# tunnel protection ipsec profile prof-blue
device(config-tnif-1)# ipv6 address 2001:DB8:4::/64
device(config-tnif-1)# exit
device(config)# ipv6 access-list 99 permit 2001:DB8:90::22/64
device(config)# route-map tunnel-route permit 99
device(config-routemap tunnel-route)# match ipv6 address 99
device(config-routemap tunnel-route)# set next-hop-ipv6-tunnel 1
device(config-routemap tunnel-route)# end
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# vrf forwarding marketing
device(config-if-e1000-1/1/3)# ipv6 policy route-map tunnel-route
device(config-if-e1000-1/1/3)# end
```

Media Access Control Security

MACsec overview

Media Access Control Security (MACsec) is a Layer 2 security technology that provides point-to-point security on Ethernet links between nodes.

NOTE
MACsec is supported on Ruckus ICX 7450, ICX 7650, and ICX 7850 devices.

MACsec, defined in the IEEE 802.1AE-2006 standard, is based on symmetric cryptographic keys. MACsec Key Agreement (MKA) protocol, defined as part of the IEEE 802.1x-2010 standard, operates at Layer 2 to generate and distribute the cryptographic keys used by the MACsec functionality installed in the hardware.

As a hop-to-hop Layer 2 security feature, MACsec can be combined with Layer 3 security technologies such as IPsec for end-to-end data security.

Supported MACsec hardware configurations

MACsec key-enabled security can be deployed on a point-to-point LAN between two connected ICX 7450, ICX 7650, or ICX 7850 devices over interfaces that share a preconfigured static key, the Connectivity Association Key (CAK).

On a licensed ICX 7450, ICX 7650, or ICX 7850 device, 10-Gbps ports can be configured for MACsec. Licenses are available per device as described in the *Ruckus FastIron Software Licensing Guide*.

NOTE
On ICX 7450 devices, MACsec is available only on 4 X 10GF modules present in slots 2, 3, or 4.

NOTE
On ICX 7650 devices, MACsec is available only on 10-Gbps fiber ports, that is, ports 25 through 48 of the base module for ICX 7650-48F devices or on slot 2 when a 4 X 10GF module is installed.

NOTE
MACsec is available on 10-Gbps ports of ICX 7850-48FS devices only.

MACsec RFCs and standards

FastIron MACsec complies with the following industry standards:

- IEEE Std 802.1X-2010: Port-Based Network Access Control
- IEEE Std 802.1AE-2006: Media Access Control (MAC) Security

- RFC 3394: Advanced Encryption Standard (AES) Key Wrap Algorithm
- RFC 5649: Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm

Refer to Port MAC Security (PMS) on page 167 for information on other IEEE 802.1X features.

MACsec considerations

Review the following considerations before deploying MACsec:

- As a prerequisite, MACsec must be licensed on each device where it is used.
- MACsec introduces an additional transit delay, due to the increase in the MAC Service Data Unit (MSDU) size.
- MACsec and Flexible authentication cannot be configured on the same port.
- On an ICX 7450 device, ports on a 4 X 10GF removable module installed in device slot 2 can be used for MACsec or stacking but not both simultaneously. In rear modules 3 and 4, MACsec can be supported at all times because stacking is not available on those modules. For more information on converting module 2 ports between MACsec and stacking, refer to the *Ruckus FastIron Stacking Configuration Guide*.

How MACsec works

MACsec capabilities prevent Layer 2 security threats, such as passive wiretapping, denial of service, intrusion, man-in-the-middle, and playback attacks.

MACsec protects communications using several configurable techniques. Data origin is authenticated and data is transported over secured channels. Frames are validated as MACsec Ethernet frames. The integrity of frame content is verified on receipt. Frame sequence is monitored using an independent replay protection counter. Invalid frames are discarded or monitored.

Data traffic carried within the MACsec frame is encrypted and decrypted using an industry-standard cipher suite.

How MACsec handles data and control traffic

All traffic is controlled on an active MACsec port; that is, data is encrypted, or its integrity is protected, or both. If a MACsec session cannot be secured, all data and control traffic is dropped.

When MACsec is active on a port, the port blocks the flow of data traffic. Data traffic is not forwarded by the port until a MACsec session is secured. If an ongoing session is torn down, traffic on the port is again blocked until a new secure session is established.

Control traffic (such as STP, LACP, or UDLD traffic) is not transmitted by an active MACsec port until a MACsec session is secured. While a session is being established, only 802.1x protocol packets are transmitted from the port. Once a secure session is established, control traffic flows normally through the port.

MACsec Key Agreement protocol

MACsec Key Agreement (MKA) protocol installed on a device relies on an IEEE 802.1X Extensible Authentication Protocol (EAP) framework to establish communication.

MACsec peers on the same LAN belong to a unique connectivity association. Members of the same connectivity association identify themselves with a shared Connectivity Association Key (CAK) and Connectivity Association Key Name (CKN). The CAK is a static key that is preconfigured on each MACsec-enabled interface. MACsec authentication is based on mutual possession and acknowledgment of the preconfigured CAK and Connectivity Association Key Name (CKN).

Each peer device establishes a single unidirectional secure channel for transmitting MACsec frames (Ethernet frames with MACsec headers that usually carry encrypted data) to its peers within the connectivity association. A connectivity association consists of two secure channels, one for inbound traffic, and one for outbound traffic. All peers within the connectivity association use the same cipher suite, either Galois/Counter Mode Advanced Encryption Standard 128 or 256 (GCM-AES-128 or GCM-AES-256), for MACsec-authenticated security functions.

MACsec Key Agreement (MKA) protocol uses the Connectivity Association Key to derive transient session keys called Secure Association Keys (SAKs). SAKs and other MKA parameters are required to sustain communication over the secure channel and to perform encryption and other MACsec security functions. SAKs, along with other essential control information, are distributed in MKA protocol control packets, also referred to as MKPDUs.

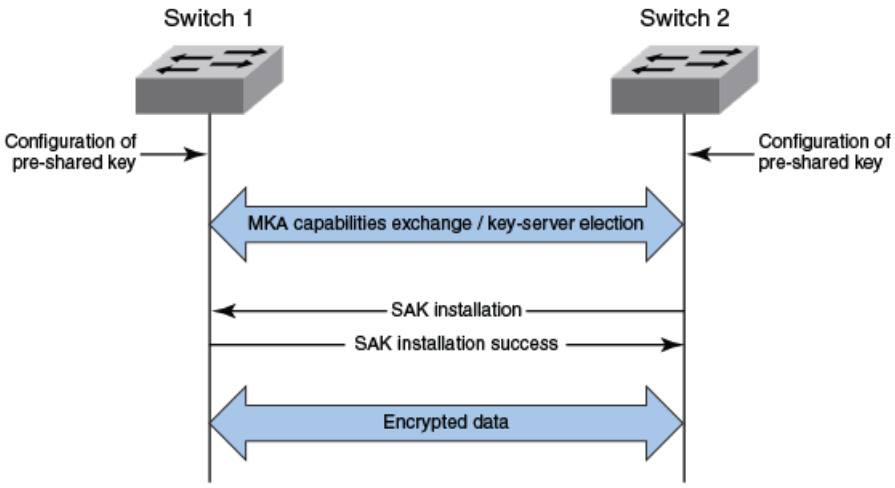
MKA message exchange between two switches

When two MACsec peers confirm possession of a shared CAK and CKN, MKA protocol initiates key-server election.

The key-server is responsible for determining whether MACsec encryption is used and what cipher suite is used to encrypt data. The key-server is also responsible for generating Secure Association Keys (SAKs) and distributing them to the connected device. Once a SAK is successfully installed, the two devices can exchange secure data.

The following figure shows the message flow between two switches during MACsec communication.

FIGURE 1 MKA pre-shared key and key name exchange between two switches



Secure channels

Communication on each secure channel takes place as a series of transient sessions called secure associations. These sessions can only be established with a unique Secure Association Key (SAK) assigned to the session.

Secure associations expire and must be re-established after transmission of a certain number of frames or after a peer disconnects and reconnects.

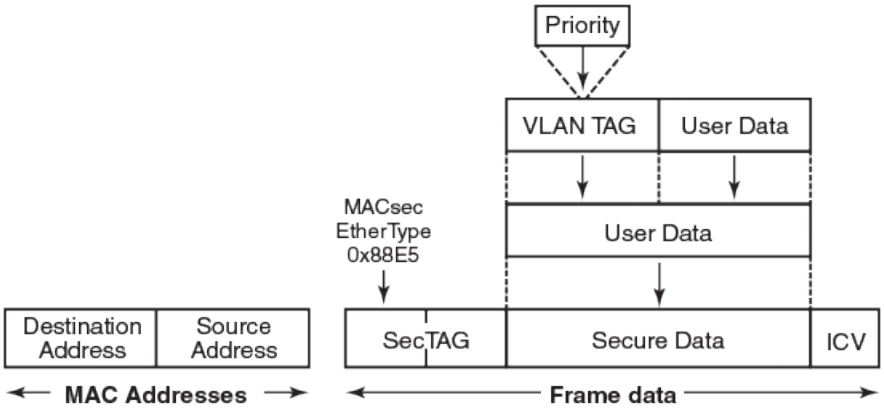
The secure association is designated by a Secure Association Identifier (SAI), formed from the Secure Channel Identifier (SCI) combined with an Association Number (AN). When a MACsec frame is received by a peer interface, the device identifies the session key from the SAI carried in the MACsec frame and uses the key to decrypt and authenticate the received frame.

MACsec frame format

When MACsec is enabled, FastIron hardware transforms each Ethernet frame by adding a security tag (secTAG) to the frame.

The following figure shows how the Ethernet frame is converted into a MACsec frame.

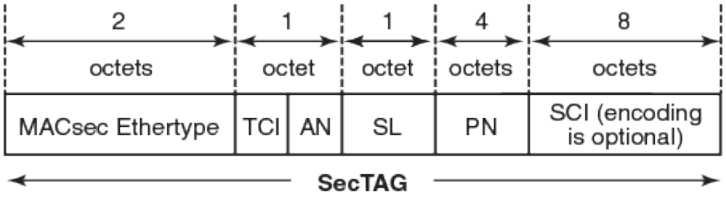
FIGURE 2 MACsec frame format



The security tag passes MACsec-related information to MACsec peers.

The following figure defines the fields in a security tag.

FIGURE 3 MACsec security tag format

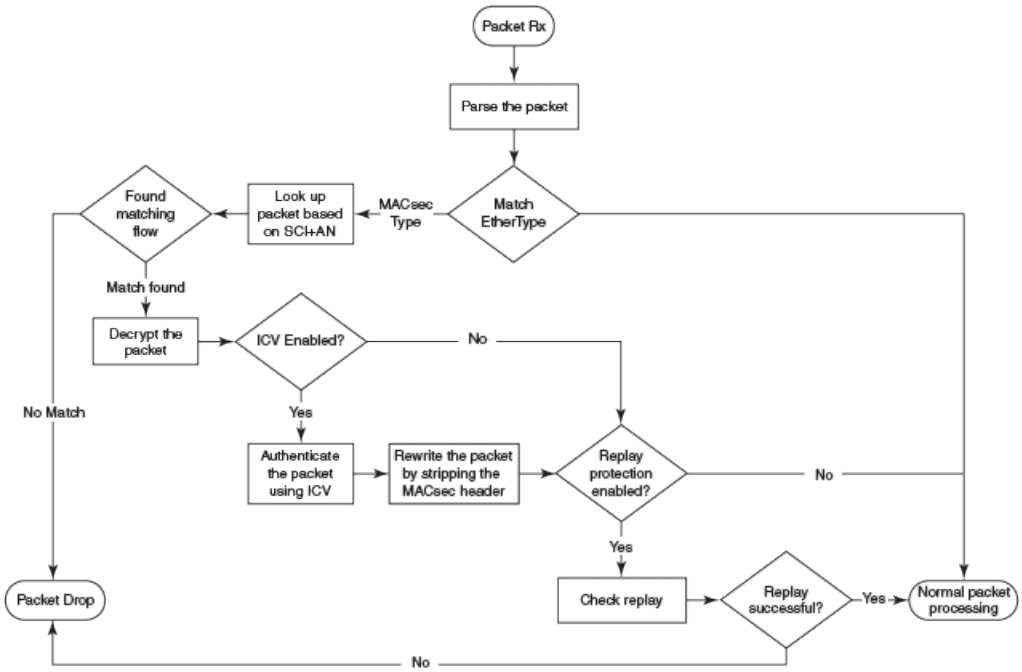


Processing incoming frames

FastIron hardware processes each MACsec frame received or transmitted based on the information in the MACsec security tag.

The FastIron device first confirms the Ethertype on incoming frames as MACsec and then processes incoming MACsec frames as illustrated in the following figure.

FIGURE 4 MACsec incoming frames

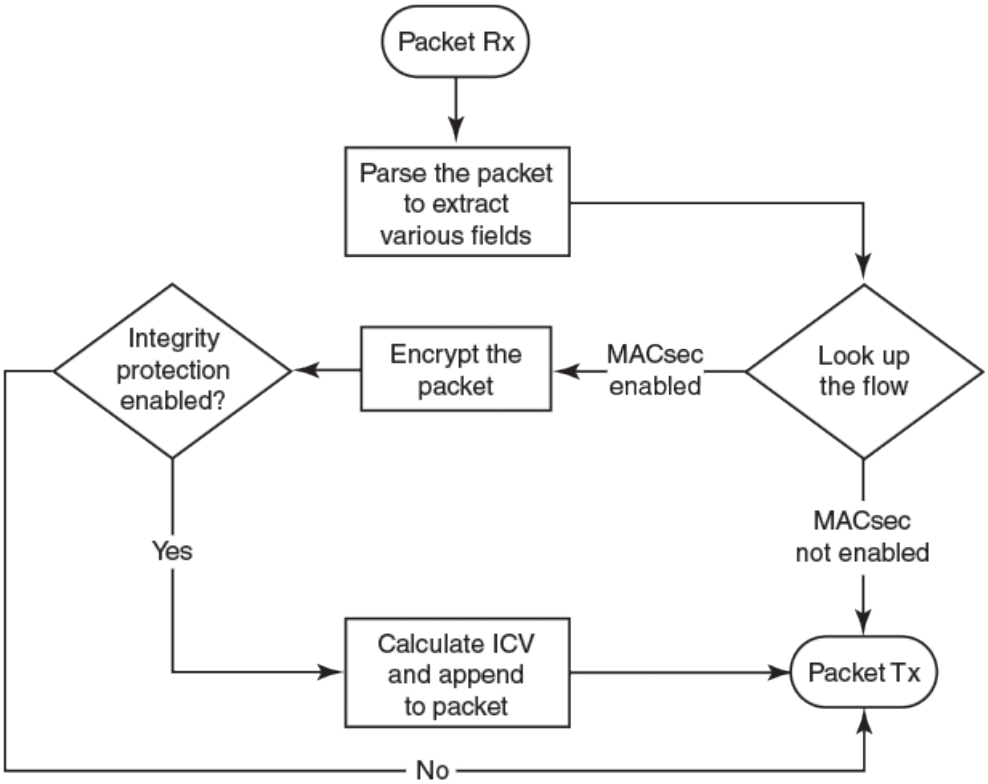


Processing outgoing frames

The FastIron device parses each outgoing frame and, if MACsec is enabled, processes the outgoing MACsec frame to apply configured MACsec options.

The following figure shows how the device applies configured MACsec options before transmitting the frames.

FIGURE 5 MACsec outgoing frames



Configuring MACsec

Although the MACsec configuration options outlined in this section are always visible, they cannot be applied unless an active license is present on the device and MACsec is enabled. MACsec licenses are required on a per-device basis. Each device in a stack requires a separate MACsec license.

These steps are required to configure MACsec security on a link or a group of connected ports.

1. Enter the dot1x-mka level from the global configuration level, and enable MACsec for the device.
2. Configure the MACsec Key Agreement (MKA) group.
3. Configure required parameters for the group, including frame validation, confidentiality, replay protection, and actions to be taken when MACsec requirements are not met.
4. Enable MKA on each participating interface.
5. Apply the configured MKA group on the participating interface.

   NOTE
   If an MKA group is not applied to an enabled MACsec interface, or if parameters within the applied group have not been configured, default values are applied to the interface. Configured parameters are visible in **show** command output; default parameters are not always visible. Refer to the *Ruckus FastIron Command Reference Guide* for default values for each command.

6. Configure the Connectivity Association Key (CAK) and Connectivity Association Key Name (CKN) on each interface.

Enabling MACsec and configuring group parameters

Enable MACsec globally on the device, and configure the MACsec Key Agreement (MKA) group before configuring MACsec security features for the group.

A valid license must be installed on the device before MACsec can be configured.

1. At the global configuration level, enter the **dot1x-mka-enable** command to enable MACsec on the device.

```
device# configure terminal
device(config)# dot1x-mka-enable
device(config-dot1x-mka)#
```

   MACsec is enabled, and the device is placed at the dot1x-mka configuration level.

   NOTE
   When MKA is disabled, all the ports are brought to a down state. You must manually enable the ports again to bring the ports back up.

2. Enter the **mka-cfg-group** command followed by a group name to create a group.

```
device# configure terminal
device(config)# dot1x-mka
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)#
```

   The group is created, and the device is placed at the group configuration level.

At the group configuration level, set the key-server priority, and define MACsec security features to be applied to interfaces once they are assigned to the group.

Configuring MACsec key-server priority

MACsec uses a key-server to generate and distribute encryption parameters and secure key information to members of a MACsec connectivity association.

The key-server is elected by comparing key-server priority values during MACsec Key Agreement (MKA) message exchange between peer devices. The elected key-server is the peer with the lowest configured key-server priority, or with the lowest Secure Channel Identifier (SCI) in case of a tie. Key-server priority may be set to a value from 0 through 255. When no priority is configured, the device defaults to a priority of 16, which is not displayed in MACsec configuration details.

NOTE
If the key-server priority is set to 255, the device will not become the key-server.

Refer to Configuring MACsec on page 156 for an overview of enabling and configuring MACsec features.

At the dot1x-mka group configuration level, enter the **key-server-priority** command, and specify a value from 0 through 255 to define the key-server priority.

```
device# configure terminal
device(config)# dot1x-mka-enable
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)# key-server-priority 20
```

In this example, the key-server priority is set to 20 for the MKA group test1.

Configure MACsec integrity and encryption for the group.

Configuring MACsec integrity and encryption

To ensure point-to-point integrity, MACsec computes an Integrity Check Value (ICV) on the entire Ethernet frame using the designated cipher suite. The designated cipher suite is also used for encryption.

MACsec adds the ICV to the frame before transmission. The receiving device recalculates the ICV and checks it against the computed value that has been added to the frame. Because the ICV is computed on the entire Ethernet frame, any modifications to the frame can be easily recognized.

By default, both encryption and integrity protection are enabled.

MACsec encrypts traffic between devices at the MAC layer and decrypts frames within participating networked devices. MACsec uses the Galois/Counter Mode Advanced Encryption Standard 128 or 256 (GCM-AES-128 or GCM-AES 256) cipher suite to encrypt data and to compute the ICV for each transmitted and received MACsec frame.

MACsec also encrypts the VLAN tag and the original Ethertype field in the Layer 2 header of the secured data. When initial bytes in a secure data packet must be transparent, a confidentiality offset of 30 or 50 bytes can be applied.

NOTE
Refer to Configuring MACsec on page 156 for an overview of enabling and configuring MACsec features.

1. At the dot1x-mka group configuration level, enter the **macsec cipher-suite** command with one of the available options:
   - gcm-aes-128: Enables encryption and integrity checking using the GCM-AES-128 cipher suite.
   - gcm-aes-128 integrity-only: Enables integrity checking without encryption.
   - gcm-aes-256: Enables encryption and integrity checking using the GCM-AES-256 cipher suite.
   - gcm-aes-256 integrity-only: Enables integrity checking without encryption.

   In the following example, MACsec 128-bit encryption has been configured as a group test1 setting. By default, the ICV integrity check is also enabled, no matter which cipher suite you use.

```
device# configure terminal
device(config)# dot1x-mka
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)# macsec cipher-suite gcm-aes-128
```

   In the following example, MACsec has been configured for integrity protection only, without encryption.

```
device# configure terminal
device(config)# dot1x-mka
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)# macsec cipher-suite gcm-aes-128 integrity-only
```

   NOTE
   The **no** form of the **macsec cipher-suite** command disables both encryption and integrity checking.

2. Enter the **macsec confidentiality-offset** command if an encryption offset is required:
   - 30: Encryption begins at byte 31 of the data packet.
   - 50: Encryption begins at byte 51 of the data packet.

   NOTE
   The default offset for MACsec encryption is zero bytes. Use the **no macsec confidentiality-offset** command to return the offset to zero bytes.

   In the following example, the encryption offset is defined as 30 bytes. The first 30 bytes of each data packet carried within the MACsec frame are transmitted without encryption.

```
device# configure terminal
device(config)# dot1x-mka
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)# macsec confidentiality-offset 30
```

Configuring MACsec frame validation

You can specify whether incoming frames are checked for MACsec (secTAG) headers and how invalid frames are handled.

NOTE
Refer to Configuring MACsec on page 156 for an overview of enabling and configuring MACsec features.

At the MKA group configuration level, enter the **macsec frame-validation** command, and select an option:

- disable: Received frames are not checked for a MACsec header.
- check: If frame validation fails, counters are incremented, but packets are accepted.
- strict: If frame validation fails, packets are dropped, and counters are incremented.

In the following example, group test1 is configured to validate frames and discard invalid ones.

```
device# configure terminal
device(config)# dot1x-mka
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)# macsec frame-validation strict
```

Configuring replay protection

MACsec replay protection detects repeated or delayed packets and acts as a safeguard against man-in-the-middle attacks.

When replay protection is configured, MACsec uses a separate replay packet number (PN) counter and gives each Ethernet frame a packet number. As frames are received, packet numbers are monitored.

Two modes of replay protection are supported: strict and out-of-order. In strict mode (the default), packets must be received in the correct incremental sequence. In out-of-order mode, packets are allowed to arrive out of sequence within a defined window.

NOTE
Refer to Configuring MACsec on page 156 for an overview of enabling and configuring MACsec features.

At the dot1x-mka group configuration level, enter the **macsec replay-protection** command with one of the available modes:

- strict: Frames must be received in exact incremental sequence.
- out-of-order *window size*: Frames are accepted out of order within the designated window size. The maximum window size is 4294967295.

- disable: Frames are not validated.

  NOTE
  The disable option is a duplicate option available on ICX 7450 devices. The **no** form of the **macsec replay-protection** command will also disable replay protection.

In the following example, replay protection is enabled for group test1. Frames must be received in exact order.

```
device# configure terminal
device(config)# dot1x-mka
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)# macsec replay-protection strict
```

In the following example, replay protection is enabled for group test1. Frames are accepted out of order within the designated window size (100).

```
device# configure terminal
device(config)# dot1x-mka
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)# macsec replay-protection out-of-order window-size 100
```

Once you have configured the desired MKA group settings, these settings can be applied to specific interfaces.

Enabling and configuring group interfaces for MACsec

After MACsec is enabled for the device, each MACsec interface must be individually enabled, and a configured group of parameters must be applied.

1. To enable MACsec, at the dot1x-mka configuration level, enter the **enable-mka** command, and specify the interface as *device/slot/port*.

   In the following example, Ethernet port 2 on slot 2 of device 1 is enabled for MACsec security.

   ```
   device# configure terminal
   device(config)# dot1x-mka
   device(config-dot1x-mka)# enable-mka ethernet 2/2/1
   device(config-dot1x-mka-2/2/1)#
   ```

   NOTE
   The following output is displayed if there is no MACsec license present on the device.

   ```
   device(config-dot1x-mka)# enable-mka ethernet 2/2/1
   Error: No MACsec License available for the port 2/2/1. Cannot enable MACsec !!!
   Error: MKA cannot be enabled on port 2/2/1
   ```

2. At the dot1x-mka interface configuration level, enter the **mka-cfg-group** command, and specify the MKA group configuration to apply to the interface.

   In the following example, MACsec options configured for group test1 are applied to the enabled interface.

   ```
   device# configure terminal
   device(config)# dot1x-mka
   device (config-dot1x-mka)# enable-mka ethernet 2/2/1
   device(config-dot1x-mka-2/2/1)# mka-cfg-group test1
   ```

Configuring the pre-shared key

MACsec security is based on a pre-shared key, the Connectivity Association Key (CAK), which you define and name. Only MACsec-enabled interfaces that are configured with the same key can communicate over secure MACsec channels.

NOTE
Refer to Configuring MACsec on page 156 for an overview of enabling and configuring MACsec features.

At the dot1x-mka-interface configuration level, enter the **pre-shared-key** command to define and name the pre-shared key.

- *Key id*: Define the key ID value using 32 hexadecimal characters.
- key-name *hex string*: Give the key a name using from 2 through 64 hexadecimal characters.

In the following example, the pre-shared key with the hex value beginning with "135bd78b" and the key name beginning with "96437a93" are applied to interface 1/3/2.

```
device# configure terminal
device(config)# dot1x-mka
device (config-dot1x-mka)# enable-mka ethernet 1/3/2
device(config-dot1x-mka-1/3/2)# pre-shared-key 135bd758b0ee5c11c55ff6ab19fdb199 key-name
96437a93ccf10d9dfe347846cce52c7d
```

Enable and configure each MACsec interface. Configure the same pre-shared key (CAK) on the interfaces between which a secure channel can be established.

Sample MACsec configuration

Here is a complete example of how to enable MACsec, configure general parameters, enable and configure interfaces, and assign a key that is shared with peers.

```
device# configure terminal
device(config)# dot1x-mka-enable

device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)# key-server-priority 5
device(config-dot1x-mka-group-test1)# macsec cipher-suite gcm-aes-128
device(config-dot1x-mka-group-test1)# macsec confidentiality-offset 30
device(config-dot1x-mka-group-test1)# macsec frame-validation strict
device(config-dot1x-mka-group-test1)# macsec replay-protection strict
device(config-dot1x-mka)# enable-mka ethernet 1/3/2

device(config-dot1x-mka-1/3/2)# mka-cfg-group test1
device(config-dot1x-mka-1/3/2)# pre-shared-key 135bd758b0ee5c11c55ff6ab19fdb199 key-name
96437a93ccf10d9dfe347846cce52c7d
device(config-dot1x-mka-1/3/2)# exit
device(config-dot1x-mka)# exit
device(config)# exit
```

Displaying MACsec information

Use MACsec **show** commands to display information on MACsec for a device, group, or individual interface. MACsec **show** commands can be used to display configuration information, to report on MACsec sessions that are currently active on a device, or to monitor MACsec statistics on a particular interface.

Displaying MACsec configuration details

You can display configuration information for all MACsec groups on a device, or you can display details for a particular group.

1. In privileged EXEC, global configuration, or dot1x-mka interface mode, use the **show dot1x-mka config** command to display MACsec configuration details for the device.

   In the following example, MACsec parameters are displayed for the device and all groups configured on it. Specific MACsec interfaces are displayed as well as the pre-shared key for each interface.

```
device(config)# show dot1x-mka config
dot1x-mka-enable
mka-cfg-group group1
  key-server-priority 20
  macsec frame-validation check
  macsec confidentiality-offset 30
  macsec cipher-suite gcm-aes-128
  macsec-replay protection out-of-order window-size 100
  enable-mka ethernet 1/3/2
mka-cfg-group group1
  pre-shared-key 135bd758b0ee5c11c55ff6ab19fd0132 key-name 96437a93ccf10d9dfe3478460cce5132
enable-mka ethernet 1/3/6
  mka-cfg-group group1
  pre-shared-key 135bd758b0ee5c11c55ff6ab19fd0132 key-name 96437a93ccf10d9dfe3478460cce51321
```

2. In privileged EXEC, global configuration, or dot1x-mka interface mode, enter the **show dot1x-mka config-group** command to display information for all configured groups. Add a group name to the command to narrow the information displayed to one group.

   The following example displays information for MKA group test1.

```
device(config)# show dot1x-mka config-group test1
 mka-cfg-group test1
  key-server-priority 5
  macsec cipher-suite gcm-aes-128 integrity-only
  macsec confidentiality-offset 30
  macsec frame-validation strict
```

   NOTE
   Group information does not include the pre-shared key or enabled connections. Use the **show dot1x-mka config** command to obtain that information.

Displaying information on current MACsec sessions

You can display MACsec session activity for an interface, including the pre-shared key name, the most recent SAI information, and a list of peers.

1. For a quick overview of current MACsec sessions, enter the **show dot1x-mka sessions brief** command in privileged EXEC, global configuration, or dot1x-mka interface mode.

```
device(config)# show dot1x-mka sessions brief

Port    Link-Status  MKA-Status  Key-Server  Negotiated Capability

1/3/2   Down         Pending     ---         ---
1/3/3   Up           Secured     No          Integrity, Confidentiality with Off. 30
1/3/4   Up           Secured     No          Integrity, Confidentiality with Off. 30
```

2. To display full details on current MACsec sessions, in privileged EXEC, global configuration, or dot1x-mka interface mode, enter the **show dot1x-mka sessions ethernet** command followed by the interface identifier.

```
device(config)# show dot1x-mka sessions ethernet 1/3/3

Interface                : 1/3/3

  MACsec Status          : Secured
  DOT1X-MKA Enabled      : Yes
  DOT1X-MKA Active       : Yes
  Key Server             : No

Configuration Status:
  Enabled                : Yes
  Capability             :  Integrity,  Confidentiality
  Desired                : Yes
  Protection             : Yes
  Frame Validation       : Disable
  Replay Protection      : Strict
  Replay Protection Size  : 0
  Cipher Suite           : GCM-AES-128
  Key Server Priority    : 20

  Local SCI              : 748ef8344a510082
  Member Identifier      : 802ed0536fcafc43407ba222
  Message Number         : 8612

Secure Channel Information:
  Latest SAK Status      : Rx & Tx
  Latest SAK AN          : 0
  Latest KI              : d08483062aa9457e7c2470e300000001
  Negotiated Capability  : Integrity, Confidentiality with offset 30

Peer Information:
State        Member Identifier          Message Number    SCI                Priority
-----        -----------------          --------------    ----------------   --------
Live         d08483062aa9457e7c2470e3             8527    748ef83443910082         20
```

Displaying MKA protocol statistics for an interface

You can display a report on MKA protocol activity for a particular interface.

In privileged EXEC, global configuration, or dot1x-mka interface mode, enter the **show dot1x-mka statistics ethernet** command to display MKA protocol statistics for the designated interface.

```
device(config-dot1x-mka-1/3/3)# show dot1x-mka statistics ethernet 1/3/3

Interface                : 1/3/3

MKA in Pkts              : 8585
MKA in SAK Pkts          : 1
MKA in Bad Pkts          : 0
MKA in Bad ICV Pkts      : 0
MKA in Mismatch Pkts     : 0
MKA out Pkts             : 8687
MKA out SAK Pkts         : 0
Number of SAK            : 1
```

Displaying MACsec secure channel activity for an interface

You can display currently enforced MACsec capabilities for a specific interface, along with secure channel statistics.

1. In privileged EXEC mode, enter the **clear macsec statistics** command for the designated interface.

   Results of the previous **show macsec statistics** command are removed.

2. In privileged EXEC, global configuration, or dot1x-mka interface mode, enter the **show macsec statistics** command to display information on MACsec configuration and secure channel activity for a particular interface.

   The following **show macsec statistics** command output is for an ICX 7450 device.

```
device# clear macsec statistics ethernet 10/2/1
device# show macsec statistics ethernet 10/2/1

Interface Statistics:
---------------------
rx Untag Pkts             : 1                   tx Untag Pkts             : 0
rx Notag Pkts             : 0                   tx TooLong Pkts           : 0
rx Badtag Pkts            : 0
rx Unknownsci Pkts        : 0
rx Nosci Pkts             : 0
rx Overrun Pkts           : 0

Transmit Secure Channels:
-------------------------

SA[0] Statistics:
Protected Pkts            : 0
Encrypted Pkts            : 4485

SA[1] Statistics:
Protected Pkts            : 0
Encrypted Pkts            : 0

SA[2] Statistics:
Protected Pkts            : 0
Encrypted Pkts            : 0

SA[3] Statistics:
Protected Pkts            : 0
Encrypted Pkts            : 0

SC Statistics:
Protected Octets          : 0                   Encrypted Octets          : 250473
Protected Pkts            : 0                   Encrypted Pkts            : 4485

Receive Secure Channels:
------------------------

SA[0] Statistics:
Ok Pkts                   : 3094                Invalid Pkts              : 0
Not using SA Pkts         : 0                   Unused Pkts               : 0
Not Valid Pkts            : 0

SA[1] Statistics:
Ok Pkts                   : 0                   Invalid Pkts              : 0
Not using SA Pkts         : 0                   Unused Pkts               : 0
Not Valid Pkts            : 0

SA[2] Statistics:
Ok Pkts                   : 0                   Invalid Pkts              : 0
Not using SA Pkts         : 0                   Unused Pkts               : 0
Not Valid Pkts            : 0

SA[3] Statistics:
Ok Pkts                   : 0                   Invalid Pkts              : 0
Not using SA Pkts         : 0                   Unused Pkts               : 0
Not Valid Pkts            : 0

SC Statistics:
OkPkts                    : 3094                Invalid Pkts              : 0
Not using SA Pkts         : 0                   Unused Pkts               : 0
Not Valid Pkts            : 0                   Unchecked Pkts            : 0
Delayed Pkts              : 0                   Late Pkts                 : 0
Valid Octets              : 0                   Decrypted Octets          : 157120
```

Port MAC Security (PMS)

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 those 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. MAC addresses that are learnt beyond the configured PMS maximum threshold value are considered as restricted MAC addresses. When a security violation occurs, an action is taken according to one of three configurable modes, as summarized in the following table.

When a security violation occurs, a Syslog entry and an SNMP trap are generated.

TABLE 16 PMS violation actions/modes

Violation Actions/Modes	Description
Protect	This is the default PMS violation action. Allows packets from secure addresses and drops all other packets. In the protect mode, the port never gets shut down. SNMP trap and Syslog are generated when the port enters and leaves the protect mode.
Restrict	Drops packets from the restricted MAC addresses and allows packets from the secure addresses. The maximum number of MAC addresses that can be restricted is 128. If the number of restricted MAC addresses exceeds 128, the port is shut down. In this mode, manual intervention is required to bring up the port that is forced to shut down after the security violation. SNMP trap and Syslog are generated for each restricted MAC address. Also, when a port is shutdown a separate trap and Syslog are generated to indicate the same.
Shutdown	Shuts down the port upon detection of first restricted MAC address. The shutdown time which serves as a recovery interval, brings up the port within a configured time without any manual intervention. SNMP trap and Syslog are generated when the port shuts down.

The secure MAC addresses are flushed when an interface is disabled and re-enabled on ICX devices. The secure addresses can be kept secure permanently (the default), or can be configured to age out, at which time they are no longer secure. You can configure the device to automatically save the secure MAC address list to the startup-config file at specified intervals, allowing addresses to be kept secure across system restarts.

Local and global resources used for port MAC security

The port MAC security feature uses a concept of local and global resources to determine how many MAC addresses can be secured on each PMS-enabled port. In this context, a resource is the ability to store one secure MAC address entry. When an interface has secured enough MAC addresses to reach its configured limit for local resources, it uses the global resources to secure additional MAC addresses.

When the port MAC security feature is enabled on an interface, the interface can store one secure MAC address (default). The maximum number of secure MAC addresses that can be secured using local resources is 64 MAC addresses per PMS-port. Besides the maximum of 64 local resources available for an interface, additional 8192 global resources are shared among all PMS-enabled interfaces on the device by default. The default system value of the global resources can be changed using the **system-max pms-global-pool** command. Global resources are shared among all the interfaces on a first-come, first-served basis. The maximum number of MAC addresses any single interface can secure is 64 (the maximum number of local resources available to the interface), plus the number of global resources not allocated to other interfaces.

Configuration considerations for port MAC security

The following limitations apply to the port MAC security (PMS) feature:

- Applies only to Ethernet interfaces.
- PMS is not supported on PVLAN ports.
- Unknown unicast traffic is flooded out of port with maximum secure MAC learnt on removing the ACL.
- Not supported on static trunk group members or ports that are configured for link aggregation.
- Not supported on 802.1X authentication-enabled ports.
- The SNMP trap generated for restricted MAC addresses indicates the VLAN ID associated with the MAC address, as well as the port number and MAC address.
- Not supported on ports that have MAC authentication enabled.
- The first packet from each new secure MAC address is dropped if secure MAC addresses are learned dynamically.
- Violated MAC movement is not supported.
- Three MAC addresses must be allocated in port security when connecting IP Phone with PC connected to it. Because, the VOIP phone initially connects untagged. After it gets its configuration and the voice VLAN from FastIron device, it connects tagged with the voice VLAN. The tagged and untagged for the same MAC address are considered as two different entries. Then the PC connected to the phone gets its MAC address registered untagged, consuming a total of three entries.

Secure MAC movement

If you move a connected device that has MAC address configured as secure on one port to another port, the FastIron device connects through the new port without waiting for the MAC address to age out on the previous port. This MAC movement feature is supported when the connected device moves from a secure port to another secure or non-secure port.

MAC movement feature is not supported in the following cases:

- MAC address is permanently secured to a port with **age 0** command.
- MAC address causes a MAC security violation on the previous port.

NOTE
Excessive Syslog messages are expected when MAC movement happens on permanently secured MAC address. Use the **no logging buffered warnings** command to suppress warning Syslogs.

Port MAC security configuration

To configure the port MAC security feature, perform the following tasks:

- Enable the port MAC security feature
- Set the maximum number of secure MAC addresses for an interface
- Set the port security age timer
- Specify secure MAC addresses
- Configure the device to automatically save secure MAC addresses to the startup-config file
- Specify the action taken when a security violation occurs

NOTE
When using the **absolute** option to age out MAC addresses on timer expiry, make sure that the age timer value is sufficient. Avoid using a very short timer expiry with the **absolute** option, as the value may be in conflict with other timer settings and may cause performance problems in the network. For example, a one-minute timer expiry will cause MAC addresses to be flushed every minute. As a result, operational (enable/disable) loops and packet flooding may occur following a security violation, which by default causes a port to be disabled for one minute.

Configuring port MAC security

Port MAC security can be configured globally or on a specific interface.

By default, the MAC port security feature is disabled on all interfaces.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter port security configuration mode.

```
device(config)# port security
```

3. Enable port MAC security globally on the device.

```
device(config-port-security)# enable
```

4. Set the maximum number of secure MAC addresses.

```
device(config-port-security)# maximum 10
```

   By default, when MAC port security is enabled, an interface can store one secure MAC address. A maximum of 64 local resources can be allocated to each interface.

5. (Optional) From the global configuration mode, configure the maximum number of global resources shared among all interfaces on the device to store secure MAC addresses.

```
device(config)# system-max pms-global-pool 800
```

   By default, 8192 global resources are shared among all interfaces on the device which is also the maximum global resources supported. The global resources are in addition to the local resources allocated to each interface . The maximum number of MAC addresses any single interface can secure is 64 (the maximum number of local resources allocated to the interface), plus the number of global resources available.

6. Specify a secure MAC address.

```
device(config-port-security)# secure-mac-address 000.0018.747c
```

7. Set the optional port security age timer. By default, learned MAC addresses stay secure indefinitely.

```
device(config-port-security)# age 60 absolute
```

   In this example, secure MAC addresses are immediately timed out after 60 minutes. If the absolute option is not configured, the secure MAC addresses time out when the hardware MAC age timer expires.

8. Set the optional time interval when learned secure MAC addresses are saved to the startup configuration.

```
device(config-port-security)# autosave 20
```

   In this example, learned secure MAC addresses are saved to the startup configuration every 20 minutes.

9. Configure the action that must be taken when a port security violation occurs. Select one of the following configurable modes that specifies the violation action.
   - Configure the protect mode to drop all packets which are not from secure MAC addresses.

```
device(config-port-security)# violation protect
```

     This is the default PMS violation action. In the protect mode, the port never gets shut down.
   - Configure the restrict mode to drop packets from violated MAC address and allow packets from secure addresses.

```
device(config-port-security)# violation restrict
```

     When the **restrict** option is used, maximum number of MAC addresses that can be restricted is 128. If the number of violated MAC addresses exceeds 128, the port will be shut down. In this mode, manual intervention is required to bring up the port that is forced to shut down after the security violation. Aging for restricted MAC addresses is done in software. There can be a worst case inaccuracy of one minute from the specified time. The restricted MAC addresses are denied in hardware.
   - Configure the shutdown mode to disable the port upon detection of first violated MAC address. for a specified amount of time, in minutes, when a security violation occurs.

```
device(config-port-security)# violation shutdown
```

     The shutdown time which serves as a recovery interval, brings up the port within a configured time without any manual intervention. The default is value is 0 which shuts down the port permanently when a security violation occurs.

In the following example, port MAC security is configured globally.

```
device# configure terminal
device(config)# port security
device(config-port-security)# enable
device(config-port-security)# maximum 10
device(config-port-security)# secure-mac-address 000.0018.747c
device(config-port-security)# age 60 absolute
device(config-port-security)# autosave 20
device(config-port-security)# violation restrict 100
```

Clearing port security statistics

You can clear restricted MAC addresses and violation statistics on all ports or from individual ports.

Clearing restricted MAC addresses

To clear all restricted MAC addresses globally, enter the **clear port security restricted-macs all** command.

```
device# clear port security restricted-macs all
```

To clear restricted MAC addresses on a specific port, enter the following command.

```
device# clear port security restricted-macs ethernet 5/1/5
```

Clearing violation statistics

To clear violation statistics globally, enter the **clear port security statistics** all command.

```
device# clear port security statistics all
```

The following command clears violation statistics on a specific port.

```
device# clear port security statistics ethernet 1/1/5
```

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 port security settings for an individual port or for all the ports on a specified module
- The secure MAC addresses configured on the device
- Port security statistics for an interface or for a module

Displaying port MAC security settings

You can display the port security settings for an individual port or for all the ports on a specified device. For example, to display the port security settings for port 1/7/11, enter the following command.

```
device# show port security ethernet 1/7/11

Port   Security Violation Shutdown-Time Age-Time Max-MAC
------ -------- --------- ------------- -------- -------
1/7/11 disabled  shutdown            10       10       1
```

Displaying secure MAC addresses information

To list the secure MAC addresses configured on the device, enter the following command.

```
device# show port security mac

Port   Num-Addr Secure-Src-Addr Resource Age-Left Shutdown/Time-Left
------ -------- --------------- -------- -------- ------------------
1/7/11        1   0000.018.747c    Local       10                 no
```

Displaying port security statistics

To display port security statistics for interface 1/7/11, enter the following command.

```
device# show port security statistics ethernet 1/7/11

Port   Total-Addrs Maximum-Addrs Violation Shutdown/Time-Left
-----  ----------- ------------- --------- ------------------
1/7/11           1             1         0                 no
```

Displaying restricted MAC addresses information

To display a list of restricted MAC addresses on a port, enter a command such as the following.

```
device# show port security ethernet 1/1/5 restricted-macs
```

Defining MAC Address Filters

MAC address filters configuration notes and limitations

- MAC address filtering on FastIron devices is performed in hardware.
- MAC address filtering on FastIron devices differ from other Ruckus devices in that you can only filter on source and destination MAC addresses. Other Ruckus devices allow you to also filter on the encapsulation type and frame type.
- MAC address filtering applies to all traffic, including management traffic. To exclude management traffic from being filtered, configure a MAC address filter that explicitly permits all traffic headed to the management MAC (destination) address. The MAC address for management traffic is always the MAC address of port 1.
- MAC address filters do not filter Layer 2 control protocols. Layer 2 control protocols, such as STP and LACP, are processed by the device even when a "Deny All" MAC address filter has been applied on the interface.
- MAC address filtering cannot be applied on management interface for all platforms.
- The number of MAC address filters that you can configure in an ICX 7150 stack is less than that in other platforms.

The following configuration notes apply to Ruckus Layer 3 devices:

- MAC address filters apply to both switched and routed traffic. If a routing protocol (for example, OSPF) is configured on an interface, the configuration must include a MAC address filter rule that allows the routing protocol MAC and the neighbor system MAC address.
- You cannot use MAC address filters to filter Layer 4 information.
- MAC address filters are supported on tagged ports in the Layer 3 software images.

MAC address filter command

MAC address filtering permit and deny traffic from devices using the specified MAC address list.

Once you apply a MAC address filter to a port, the device drops all ethernet traffic on the port that does not match a MAC permit filter on the port. The permit any statement allows all MAC addresses that are not specified in the MAC filter-group. The filters must be applied as a group. For example, if you want to apply four filters to an interface, they must all appear on the same command line. You cannot add or remove individual filters in the group. To add or remove a filter on an interface, apply the filter group again containing all the filters you want to apply to the port. If you apply a filter group to a port that already has a filter group applied, the older filter group is replaced by the new filter group.

The following steps configure the MAC address filtering based on the MAC address list.

NOTE
Each ICX device supports a maximum of 512 MAC filters per system and the default value is 64. The maximum value of MAC filters per port is 256 and the default value is 32 per port.

1. Enter global configuration mode.

```
device# configure terminal
device(config)#
```

2. The below commands define the list of MAC address filters to permit or deny traffic based on source and/or destination mac-address.

```
device(config)# mac filter 1 deny 0010.0075.3676 ffff.0000.0000
device(config)# mac filter 2 deny any 0000.0023.fbcd ffff.ffff.ffff
device(config)# mac filter 3 deny any 0180.c200.0000 ffff.ffff.fff0
device(config)# mac filter 4 deny any 0000.0034.5678 ffff.ffff.ffff
device(config)# mac filter 5 deny any 0000.0045.6789 ffff.ffff.ffff
device(config)# mac filter 1024 permit any any
```

   In source or destination mac-address field, you can specify a particular mac-address with exact mask or a range of mac-addresses with a variable mask or specify **any** to match any mac-address. Specify the mask using f (ones) and zeros. For example, to match on the first two bytes of the address 0010.0075.3676, use the mask ffff.0000.0000. In this case, the filter matches on all MAC addresses that contain "0010" as the first two bytes. The filter accepts any value for the remaining bytes of the MAC address.

3. Apply previously created MAC filters to an interface using the mac filter-group command.

```
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# mac filter-group 1 to 5 1024
```

   When applying the filter-group to the interface, specify each filter-id to be applied separately or specify a range of filter-ids, for example, 1 3 to 8 10.

The following example denies the Layer 2 traffic if the conditions of filter lines 1 to 5 are matched, and permit the traffic from all other source mac address as per the last statement or filter number 1024. These filter lines from 1 to 5 and the permit command 1024 is then applied to a specific port 1/1/2.

```
device# configure terminal
device(config)# mac filter 1 deny 0010.0075.3676 ffff.0000.0000
device(config)# mac filter 2 deny any 0000.0023.fbcd ffff.ffff.ffff
device(config)# mac filter 3 deny any 0180.c200.0000 ffff.ffff.fff0
device(config)# mac filter 4 deny any 0000.0034.5678 ffff.ffff.ffff
device(config)# mac filter 5 deny any 0000.0045.6789 ffff.ffff.ffff
device(config)# mac filter 1024 permit any any
device(config)# interface ethernet 1/1/2
device(config-if-e1000-1/1/2)# mac filter-group 1 to 5 1024
```

When a MAC address filter is applied to or removed from an interface, a Syslog message is generated.

```
SYSLOG: <14>Jan 1 00:00:00 10.44.9.11 MAC Filter applied to port 1/1/2 by tester from telnet session
(filter id=5)
SYSLOG: <14>Jan 1 00:00:00 10.44.9.11 MAC Filter removed from port 1/1/2 by tester from telnet session
(filter id=5)
```

The Syslog messages indicate that a MAC address filter was applied to the specified port by the specified user during the specified session type. Session type can be Console, Telnet, SSH, Web, SNMP, or others. The filter IDs that were added or removed are listed.

Displaying MAC address filter information

To display the maximum number of MAC filters allowed per system and per interface or to display the configured MAC filter groups.

Use the **show l2 filter-list** command in global configuration mode to display the information of all configured MAC filters.

```
device# show l2 filter-list
max_number_of_l2_filters           : 64
max_number_of_l2_filters_per_table : 32
mac filter 1 deny  0033.1144.2201 ffff.ffff.ffff any
mac filter 2  permit any  any
mac filter 44 permit 0000.0004.0001 ffff.ffff.ffff 0000.0001.0004 ffff.ffff.ffff
mac filter 61 deny  0000.0001.0001 ffff.ffff.ffff 0000.0001.0002 ffff.ffff.ffff
mac filter 62 deny  0000.0002.0001 ffff.ffff.ffff ffff.ffff.ffff ffff.ffff.ffff
```

Enabling logging of management traffic permitted by MAC address filters

You can configure the Ruckus device to generate Syslog entries and SNMP traps for management traffic that is permitted by MAC address filters. Management traffic applies to packets that are destined for the CPU, such as control packets. You can enable logging of permitted management traffic on a global basis or an individual port basis.

The first time an entry in a MAC address filter permits a management packet and logging is enabled for that entry, the software generates a Syslog message and an SNMP trap. Messages for management packets permitted by MAC address filters are at the warning level of the Syslog.

When the first Syslog entry for a management packet permitted by a MAC address filter is generated, the software starts a five-minute timer. After this, the software sends Syslog messages every five minutes. The messages list the number of management packets permitted by each MAC address filter during the previous five-minute interval. If a MAC address filter does not permit any packets during the five-minute interval, the software does not generate a Syslog entry for that MAC address filter.

> **NOTE**
> For a MAC address filter to be eligible to generate a Syslog entry for permitted management packets, logging must be enabled for the filter. The Syslog contains entries only for the MAC address filters that permit packets and have logging enabled.

When the software places the first entry in the log, the software also starts the five-minute timer for subsequent log entries. Thus, five minutes after the first log entry, the software generates another log entry and SNMP trap for permitted management packets.

MAC address filter logging

MAC address filter logging enables the log entry of permitted management traffic globally or on specified port basis.

To configure MAC address filter logging globally, enter the following CLI commands at the global CONFIG level.

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Specify the port.

   ```
   device(config)# interface ethernet 1/1/1
   ```

3. Enables logging for filtered packets on a specific port.

```
device(config-if-e1000-1/1/1)# mac filter-group log-enable
```

4. Save the configuration.

```
device(config)# write memory
```

The following example enables the MAC address filter logging for MAC address filters applied to ports 1/1/1 and 1/1/3.

```
device# configure terminal
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# mac filter-group log-enable
device(config-if-e1000-1/1/1)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# mac filter-group log-enable
device(config)# write memory
```

Configuring MAC filter accounting

Steps to configure and display Layer 2 MAC filter accounting

For FastIron devices, ACL accounting is supported on Layer 2 MAC filters.

1. To enable ACL accounting on a Layer 2 MAC filter, use the **mac filter** in the global configuration mode.
2. To display MAC accounting information, use the **show access list accounting** command. The accounting statistics is collected every five seconds and is synchronized to remote unit(s) every one minute.

```
device#show access-list accounting ethernet 3/1/2 in

MAC Filters Accounting Information
    0: DA ANY SA 0000.0000.0001 - MASK FFFF.FFFF.FFFF
       action to take : DENY
     Hit Count:   (1Min)                      0    (5Sec)      0
                  (PktCnt)                    0  (ByteCnt)     0
    ------------------------------------------------------
65535: Implicit Rule deny any any
     Hit Count:   (1Min)           5028   (5Sec)    2129
                  (PktCnt)         5028 (ByteCnt)  643584
    ------------------------------------------------------
```

3. To clear ACL accounting statistics for ACLs configured, choose one of the following options.
   - For ACLs configured on a specific interface, use the **clear access list accounting** command in the global configuration mode.
   - For all ACLs configured in the device, use the **clear access list accounting all** command in the global configuration mode.

```
device(config)#clear access-list accounting ethernet 1/1/5 in

device(config)#clear access list accounting all
```

The following example shows MAC filter "10" on which ACL accounting is enabled.

```
device(config)#mac filter 10 enable-accounting
```

MAC address filter override for 802.1X-enabled ports

The MAC address filtering feature on an 802.1X-enabled port allows 802.1X and non-802.1X devices to share the same physical port. For example, this feature enables you to connect a PC and a non-802.1X device, such as a Voice Over IP (VOIP) phone, to the same 802.1X-enabled port on the Ruckus device. The IP phone will bypass 802.1X authentication and the PC will require 802.1X authentication.

To enable this feature, first create a MAC address filter, then bind it to an interface on which 802.1X is enabled. The MAC address filter includes a mask that can match on any number of bytes in the MAC address. The mask can eliminate the need to enter MAC addresses for all non-802.1X devices connected to the Ruckus device, and the ports to which these devices are connected.

MAC address filter override configuration notes

- This feature is supported on untagged, tagged, and dual-mode ports.
- You can configure this feature on ports that have ACLs and MAC address filters defined.

Configuring MAC address filter override

The dot1x auth-filter command binds the MAC address filters to a port.

To configure MAC address filter override on an 802.1X-enabled port, follow these steps.

1. Enter the dot1x configuration mode.
2. Enter the specific interface configuration and enter the **dot1x auth-filter** command followed by the parameters *id* and *vlan*.

   The example shows configuring MAC address filter override.

```
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# dot1x auth-filter <id> vlan <vlan>
```

Flexible Authentication

Flexible authentication overview

Flexible authentication combines MAC authentication and 802.1X authentication as a single authentication procedure.

NOTE
Flexible authentication is not supported in private VLANs.

NOTE
The term "client" is used to indicate the user or device that is going through authentication.

In a network, many types of clients may gain access and use the network resources. Such networks cannot be left unrestricted due to security concerns. There must be a mechanism to enforce authentication of the clients before allowing access to the network. In addition, a single authentication method may not be compatible for all the clients that support different authentication methods. In such cases, it is not feasible to assign separate ports with specific authentication methods for different types of clients. 802.1X authentication and MAC authentication, and a combination of both, provide strong, yet flexible methods to validate the clients and prevent unauthorized clients from gaining access to the network. If the authentication succeeds, the client (the MAC address of the client) is moved to a VLAN returned by the RADIUS server, and the policies returned by the RADIUS server are applied.

Ruckus ICX devices support the IEEE 802.1X standard for authenticating clients attached to data ports. Using 802.1X, you can configure an ICX device to grant or deny access to a port based on information supplied by a client to an authentication server.

When a user logs in to a network that uses 802.1X, the Ruckus ICX device grants or denies access to network services after the user is authenticated by an authentication server. The user-based authentication in 802.1X provides an alternative to granting network access based on a user IP address, MAC address, or subnetwork.

MAC authentication is another mechanism by which incoming traffic originating from a specific MAC address is switched or forwarded by the device only if an authentication server successfully authenticates the source MAC address. The MAC address itself is used as the username and password for authentication; the user does not need to provide a specific username and password to gain access to the network. If authentication for the MAC address is successful, traffic from the MAC address is forwarded in hardware.

Flexible authentication provides a means to set the sequential order in which 802.1X authentication and MAC authentication methods need to be executed. If both authentication methods are enabled on the same port, by default, the authentication sequence is set to perform 802.1X authentication followed by MAC authentication. Both the 802.1X authentication and MAC

authentication methods must be enabled at the global and interface levels on the same port to execute Flexible authentication. Flexible authentication facilitates multiple authentication methods to validate a client using a single configuration on the same port. Thus, different clients that support different types of authentication can be authenticated using a single configuration.

After successful authentication, different policies can be applied to restrict the way the client access network resources. VLAN policies, phone policies, and ACL policies can be enforced using VLAN assignment, ACL assignment, and phone-specific information to provide different levels of service to the client and to control the destination of the client.

The following table shows ICX features that have been tested for compatibility with various Network Access Control (NAC) applications.

TABLE 17 Features tested for compatibility with various NAC applications

Feature	Free RADIUS	Aruba ClearPass	Ruckus Cloudpath	Cisco ISE
802.1X authentication	Yes	Yes	Yes	Yes
MAC authentication	Yes	Yes	Yes	Yes
Dynamic VLAN assignment	Yes	Yes	Yes	Yes
Dynamic ACL assignment	Yes	Yes	Yes	Yes
External web authentication	Not applicable	Yes	Yes	Yes (Release 2.1)
Change of Authorization (CoA)	Yes	Yes	Yes	Yes (Release 2.1)

NOTE
Refer to the *Ruckus FastIron Features and Standards Support Matrix* for the list of supported platforms.

MAC VLANs

Traditional VLANs associate ports as untagged or tagged. A port can only belong to a single VLAN as untagged, yet it can be part of multiple VLANs as a tagged member.

Packets received at the port are classified into VLANs based on the VLAN tag carried in the packet (tagged VLANs); otherwise, the packet is classified with port untagged VLAN.

Using a MAC VLAN is a way of classifying the packets based on the source MAC address with the help of hardware maintaining the MAC-VLAN table.

After successful authentication, VLANs are dynamically assigned based on the client profiles configured on the RADIUS server. The ICX device then associates the port with the dynamic VLAN only for the specific client MAC address. With this option, a port can belong to multiple VLANs as a MAC VLAN member. All such packets coming from the respective clients are untagged and are classified into appropriate VLANs when they are received on the ICX device. This makes the port look as if it is part of multiple untagged VLANs. In summary, after successful authentication, the RADIUS server returns the details of the VLAN where the client should belong. The client (the MAC address of the client) is moved to this VLAN as a MAC VLAN member. The client is removed from the corresponding VLAN when the client logs out, the port goes down, or the MAC address ages out.

Data VLAN requirements for Flexible authentication

For deploying Flexible authentication, VLANs such as the auth-default VLAN, restricted VLAN, critical VLAN, and guest VLAN are used for various success, failure, and timeout scenarios. The use of these VLANs provides network administrators more granular access control for different client scenarios.

Before authentication is enabled on a port, the port can belong to any VLAN, including the system default VLAN. The only restriction is that the port cannot be a part of any VLAN as untagged. After authentication is enabled on that port, the port becomes a part of the auth-default VLAN. When a VLAN is assigned after successful authentication, it is assigned to the client (to the MAC address of the client), not to the entire port. In reality, however, the port is added to the VLAN as a MAC-VLAN member.

When authentication succeeds, the client is moved to the VLAN returned by the RADIUS server. When the Radius authentication fails or the Radius server is unavailable, the client is moved to the restricted or critical VLAN.

NOTE
A system default VLAN, reserved VLAN, or VLAN group cannot be used as the auth-default VLAN, RADIUS-assigned VLAN, restricted VLAN, critical VLAN, or guest VLAN. A system default VLAN cannot be used for Web authentication.

NOTE
A system-default VLAN, restricted VLAN, critical VLAN, or guest VLAN can’t be used as a RADIUS assigned VLAN.

You can also configure specific VLANs to associate the clients in various success, failure, and timeout scenarios. The following scenarios and options are available to place the client in various VLANs depending on the authentication status:

- Auth-default VLAN: A VLAN must be configured as the auth-default VLAN to enable Flexible authentication. When any port is enabled for 802.1X authentication or MAC authentication, the client is moved to this VLAN by default. The auth-default VLAN is also used in the following scenarios:
  - When the RADIUS server does not return any VLAN information upon authentication, the client is authenticated and remains in the auth-default VLAN.
  - If RADIUS timeout occurs during the first authentication and the timeout action is configured as "Success", the client is authenticated in the auth-default VLAN. If the RADIUS server is not available during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN.
- Restricted VLAN: When an authentication fails, the port can be moved into a restricted VLAN instead of blocking the client completely. The port is moved to the configured restricted VLAN only if the authentication failure action is configured to place the port in a restricted VLAN using the **auth-fail-action** command in the authentication configuration mode. Otherwise, when the authentication fails, the client's MAC address is blocked in the hardware, which is the default action. A restricted VLAN can be configured using the **restricted-vlan** command in the authentication configuration mode.
- Critical VLAN: There may be times when the RADIUS server times out or is not available, resulting in timeout. This can happen the first time the client is authenticating or when the client reauthenticates. In such scenarios, if the authentication timeout action is specified as a critical VLAN using the **authentication timeout-action** command in the authentication configuration mode, the client is moved to the specified critical VLAN. A critical VLAN can be configured using the **critical-vlan** command in the authentication configuration mode.
- Guest VLAN: The guest VLAN is used when a client does not respond to dot1x requests for authentication. It is possible that the client does not support or have the dot1x supplicant loaded. In such a scenario, the client is moved to the guest VLAN to have access to the network with default privileges. From the guest VLAN, the client can download the supplicant.

  NOTE
  The same VLAN can be specified as guest, restricted, and critical VLAN.

Voice VLAN requirements for Flexible authentication

Voice VLAN configuration facilitates the continued functioning of Voice over IP (VoIP) phones when external server authentication or authorization fails.

A voice VLAN is configured at the global level (using the **voice-vlan** command in authentication configuration mode) or at the local level (using the **authentication voice-vlan** command in interface configuration mode).

When a local voice VLAN is configured, it overrides the global voice VLAN configuration. The global voice VLAN is only used when a voice VLAN is not configured on the port.

A voice VLAN is used for voice communication by IP phones in the following ways:

- Authentication Success: When a RADIUS server authenticates an IP phone and the server does not specify a VLAN, the device is placed in the auth-default VLAN and also added to the voice VLAN (as a tagged member). The authdefault VLAN is used for initial authentication, learning the IP address, and so on, while the voice VLAN is used for voice traffic.
- Authentication Failure: When a phone fails authentication by a RADIUS server, the phone is blocked in hardware by default. To ensure that an IP phone continues to operate and that both data and voice traffic from the IP phone are appropriately forwarded, use the **auth-fail-action** command to set the fail action to **restricted-vlan**, and specify the **voice voice-vlan** option. This causes the device to be placed in the restricted VLAN for data traffic and the voice VLAN (as a tagged member) for voice traffic.
- Authentication Timeout: When a RADIUS server is not reachable, the phone cannot be authenticated, in which case several **timeout-action** options are available: treat as success, treat as failure, and treat as critical. Success and Failure cases work as explained previously. The critical-VLAN case works in a similar way to the restricted-VLAN case. To ensure that an IP phone continues to operate and that both data and voice traffic from the IP phone are appropriately forwarded, use the **auth-timeout-action** command to set the timeout action to **critical-vlan**, and specify the **voice voice-vlan** option. This causes the device to be placed in the critical VLAN for data traffic and the voice VLAN (as a tagged member) for voice traffic.

Authentication modes

Flexible authentication supports the following four modes to address the requirements of different usage models. The authentication-enabled ports can be configured to be a member of any mode using the **auth-mode** command at the global level or using the **authentication auth-mode** command from the interface.

- Multiple Untagged: Different clients on the same port can be placed in different untagged VLANs. Each client can be assigned a different VLAN by the RADIUS server. Some clients who fail can move to restricted-VLAN. Some clients who time out out can move to guest and/or other VLANs.
- Single Untagged: This is the default auth-mode, where all the clients belong to one untagged VLAN only. This mode is the most common use case. When a hub and multiple clients are connected, the first client is moved to the RADIUS-assigned VLAN, and the subsequent clients are placed in the same VLAN. The subsequent authenticated clients are moved to the same VLAN even if RADIUS does not return any VLAN. If RADIUS returns different untagged VLANs, subsequent clients are blocked.
- Single Host: This mode allows authentication of only one host, and the status of the host is determined by the authentication. Access for all subsequent hosts is denied or blocked. In contrast, any connected IP phones are allowed access without authentication. The session exists only for the first host authenticated.
- Multiple Hosts: This mode allows authentication of the first device only, and the status of all other devices depends on the authentication status of the first device. This mode is typically used when the connecting device is a wireless AP that is authenticated by the ICX device, and the AP authenticates all pass-through wireless clients on the ICX device.

Dynamic assignment of VLANs in these modes varies depending on the format of the VLAN information in the RADIUS-returned Access-Accept message and whether the authentication is initiated by tagged or untagged ports. For more information, refer to Dynamic VLAN assignment on page 205

Tagged VM client support

Flexible authentication supports the authentication of VM clients that operate in tagged VLANs. When the port is tagged, the connected VMs are placed in the VM-tagged VLANs as expected. This arrangement requires the administrator to configure the ports to be tagged and to plan the network ahead of time, and the approach becomes static and inflexible for dynamic network usage changes.

The best alternative is to allow the tagged traffic from VMs to trigger authentication. When the port is not tagged, authenticate it, and dynamically tag the port in the required tagged VLANs as long as the VMs are in use. When the VM sessions on the ICX device expire for various reasons, the port is deleted from the respective tagged VLANs. This support is disabled by default and can be enabled at the interface level with the **authentication allow-tagged configuration** command. This mode works only at the interface level and must be applied on all the interfaces where it is required.

Static authentication with MAC filters

Some devices may always need to be authenticated (whitelist), and some may always have to be blocked (blacklist). Device disposition can be determined at the time of authentication through the RADIUS server; however, for convenience, MAC address filters (auth-filters) can be configured on ICX devices and applied to appropriate ports so that the ports are pre-authenticated (excluded from authentication based on MAC address).

MAC address filter must be used when the RADIUS server itself is connected to an interface on which MAC authentication or 802.1X authentication is enabled. If a MAC address filter is not defined for the MAC addresses of the RADIUS server and applied on the interface, the RADIUS authentication process fails because the device drops all packets from the RADIUS server itself.

The MAC address filter is applied on an interface using the **authentication auth-filter** command in the interface configuration mode. A client can be authenticated in an untagged VLAN or tagged VLAN. If the MAC address filter has a tagged VLAN configuration, the clients are authenticated in the auth-default VLAN and the tagged VLAN provided in the MAC address filter. The clients authorized in the auth-default VLAN allow both untagged and tagged traffic.

Authentication actions

With Flexible authentication, success, failure, and timeout actions are applied for MAC authentication and 802.1X authentication.

Authentication success action

When the authentication order is set to perform 802.1X authentication followed by MAC authentication (the default Flexible authentication sequence), on 802.1X authentication, the client is authenticated, and the policies returned by the RADIUS server are applied.

When the authentication order is set to perform MAC authentication followed by 802.1X authentication, by default, 802.1X authentication is performed even if MAC authentication is successful. On successful 802.1X authentication, the client is authenticated, and the policies returned by the RADIUS server are applied.

Authentication failure action

NOTE
The RADIUS server cannot dynamically assign critical, restricted, guest, or default VLANs.

A single failure action can be defined for both 802.1X authentication and MAC authentication. An administrator can take the following actions when there is an authentication failure:

- Block the client access (the default action): This blocks the client from accessing any network resource for a configured amount of time, after which the client can try authenticating again.
- Move the client to a restricted VLAN: This will happen only when the authentication server sends ACCESS-REJECT. It moves the client to a preconfigured restricted VLAN. Any access policies applied in that VLAN apply to this client.

Authentication timeout action

A single authentication timeout action can be specified for MAC authentication and 802.1X authentication timeouts with the RADIUS server.

A RADIUS timeout occurs when the ICX device does not receive a response from a RADIUS server within a specified time and after a certain number of retries. The time limit and number of retries can be manually configured using the **radius-server timeout** and **radius-server retransmit** commands. If the parameters are not manually configured, the ICX device applies the default value of 3 seconds with a maximum of 3 retries.

Administrators can control port behavior when a RADIUS timeout occurs by configuring a port on the ICX device to automatically pass or fail user authentication. A pass allows the client to continue with the VLAN and other policies. A fail blocks the client by default, unless a restricted VLAN or a default ACL is configured, in which case, the user is placed into a VLAN.

The following options are available:

- Failure (the default): This action blocks the client from accessing any network resource for a configured amount of time. If the failure action is configured as a restricted VLAN, the client is moved to the restricted VLAN.
- Success: The client is authenticated in the auth-default VLAN or in the previously authenticated VLAN, depending on the following conditions:
  - If RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the auth-default VLAN.
  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN.
  - If the RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the critical VLAN.
  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN.
- Critical VLAN: The client is moved to a preconfigured critical VLAN. Any access policies applicable to that VLAN apply to this client.

  NOTE
  The RADIUS server cannot dynamically assign critical, restricted, guest, or default VLANs.

  NOTE
  Reauthentication is supported for restricted and critical VLANs. It is not supported for guest VLANs.

  NOTE
  The same VLAN can be specified as guest, restricted, and critical VLAN, if desired.

Reauthentication for the clients placed in a critical VLAN and the auth-default VLAN can be configured using the **authentication reauth-timeout** command at the global level. The timeout is enabled by default and is set to 300 seconds.

Session limits on an interface

Flexible authentication allows multiple MAC addresses to be authenticated or denied on each interface.

By default, the number of MAC sessions that can be authenticated on a single interface is two. The maximum number of sessions can be changed using the **authentication max-sessions** command. The maximum number of authenticated MAC sessions on an interface depends on the ICX device and dynamic ACL assignments. If RADIUS assigns dynamic ACLs to at least one client on the interface, the maximum number of MAC sessions that can be authenticated is limited to 32 in all ICX devices.

If a dynamic ACL is not assigned to any of the clients on the interface, the maximum number of MAC addresses that can be authenticated varies depending on the ICX device as specified in the following table. System reload is not required for the changes to take effect. However, existing sessions on the interface are cleared for the changes to take effect.

TABLE 18 Maximum number of authenticated MAC sessions per port by ICX device

Supported platforms	Maximum number of MAC sessions per port when none of the clients has dynamic ACL	Maximum number of MAC sessions per port when at least one client has dynamic ACL
ICX 7850	1024	32
ICX 7750	1024	32
ICX 7650	1024	32
ICX 7450	1024	32
ICX 7250	1024	32
ICX 7150	1024	32

The system limit for authenticated MAC sessions also varies and depends on the ICX device and dynamic ACL assignments.

TABLE 19 Maximum number of authenticated MAC sessions per system (standalone or stack) by ICX device

Supported platforms	Maximum number of MAC sessions per system when none of the clients has dynamic ACL	Maximum number of MAC sessions per system when at least one client has dynamic ACL
ICX 7850	1536	512
ICX 7750	1536	512
ICX 7650	1536	512
ICX 7450	1536	512
ICX 7250	1536	512
ICX 7150	1536	512

How Flexible authentication works

Flexible authentication can be configured at the global or interface level.

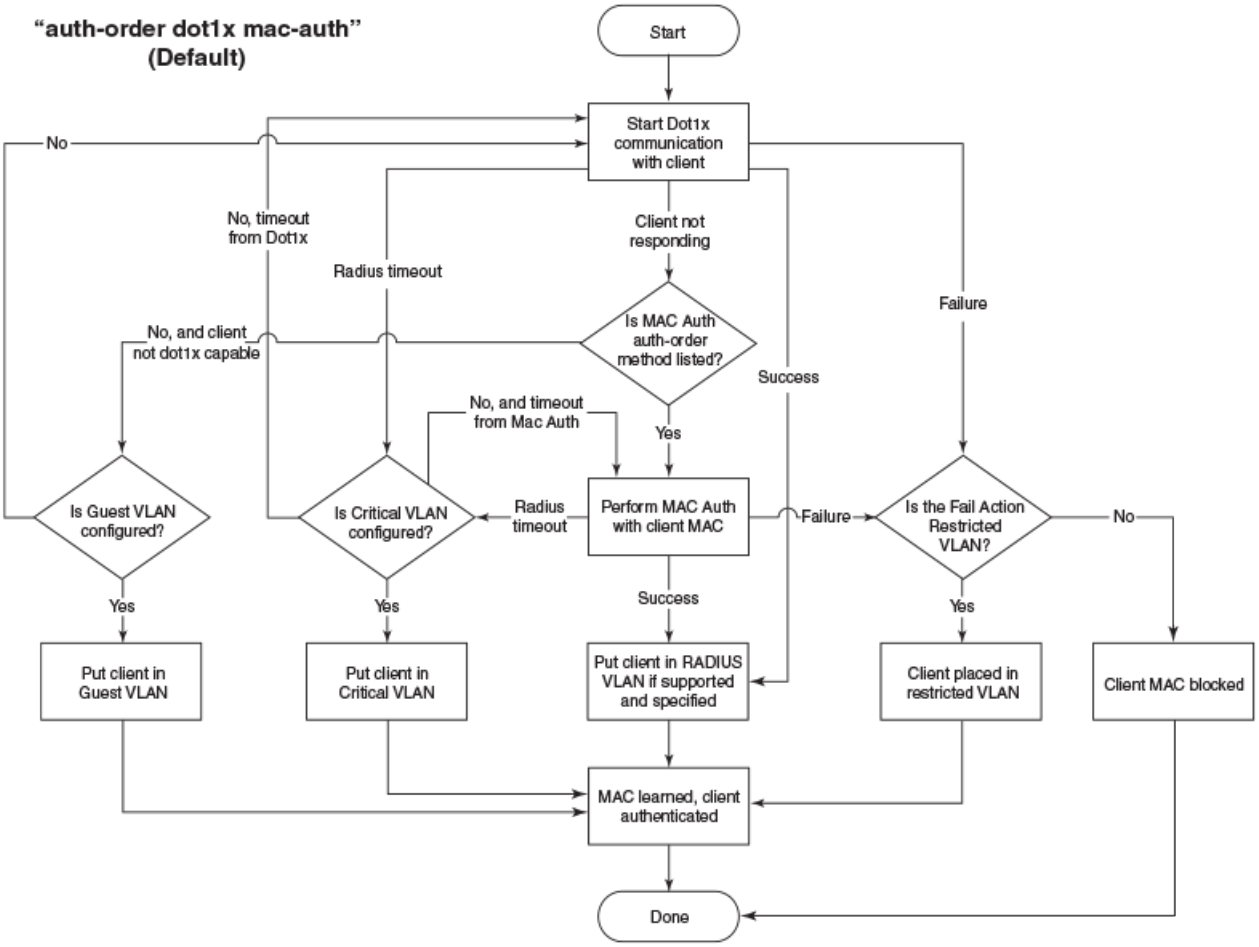
When only 802.1X authentication or MAC authentication is configured, the configured method is attempted. When authentication fails, the MAC address of the device is blocked (the default action) or is moved to a restricted VLAN when configured on the switch as the authentication failure action. If authentication succeeds, the client is authenticated, and the policies returned by the RADIUS server are applied. When both 802.1X and MAC authentication methods are configured, authentication is performed depending on the authentication order, as explained in the following sections.

Authentication order: 802.1X authentication followed by MAC authentication

When the 802.1X authentication and MAC authentication methods are enabled on the same port, the default authentication order is to perform 802.1X authentication followed by MAC authentication (refer to Figure 6).

When 802.1X authentication succeeds, the client is authenticated, and the policies returned by the RADIUS server are applied. MAC authentication is not performed in this case. If 802.1X authentication fails and MAC authentication override is configured, MAC authentication is attempted; otherwise, the failure action is carried out. On the other hand, if the client does not respond to dot1x messages, MAC authentication is attempted after the client is declared non-dot1x capable. On successful MAC authentication, the client is authenticated, and the policies returned by the RADIUS server are applied. On authentication failure, the configured failure action is applied.

FIGURE 6 Authentication sequence: 802.1X authentication followed by MAC authentication



Authentication order: MAC authentication followed by 802.1X authentication

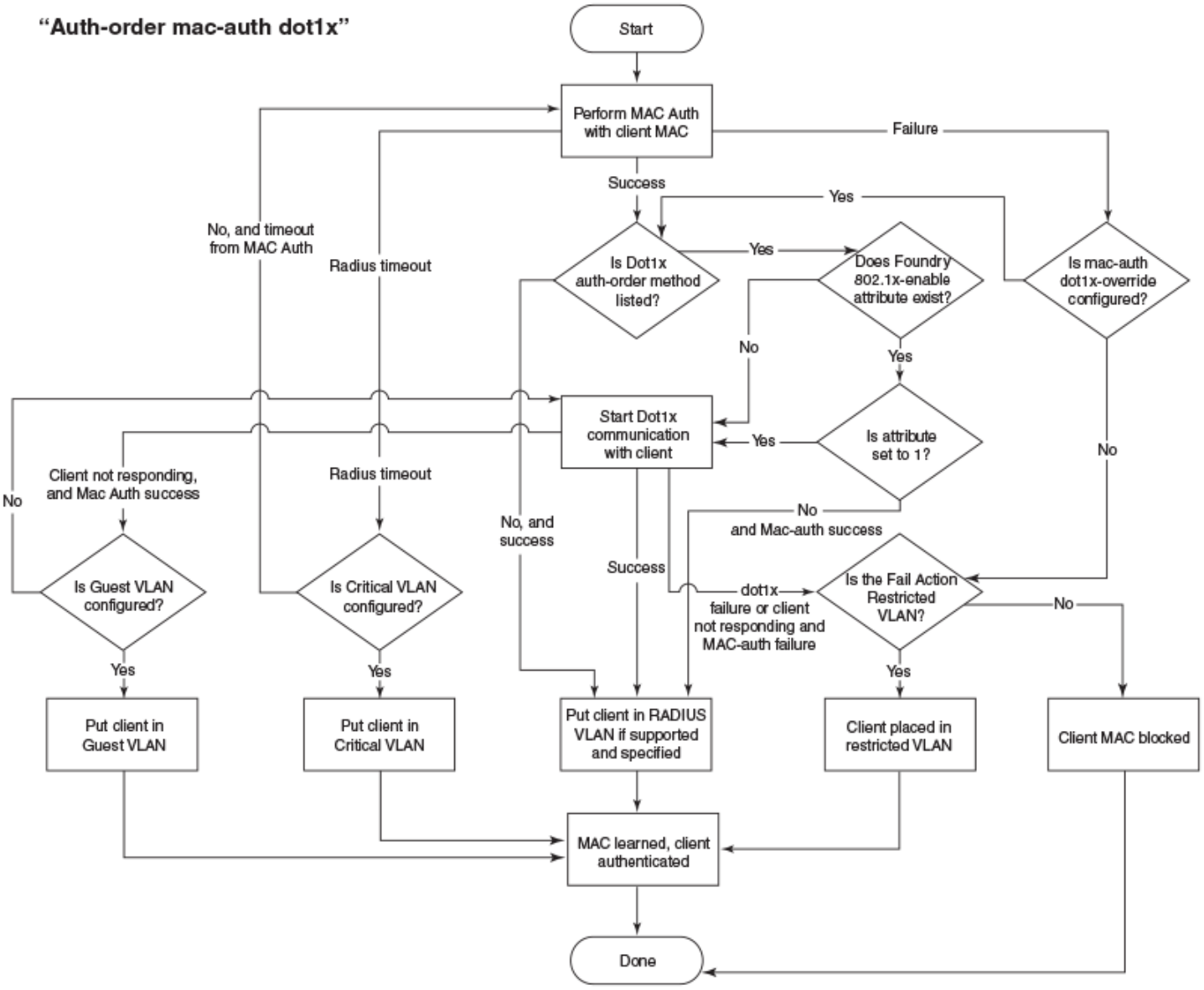
When the authentication order is set to perform MAC authentication followed by 802.1X authentication, by default, 802.1X authentication is performed, even if MAC authentication is successful (refer to Figure 7). On successful 802.1X authentication, the client is authenticated, and the policies returned by the RADIUS server are applied. On authentication failure, the configured failure action is applied.

The default behavior can be changed by specifying the RADIUS attribute (refer to Company-specific attributes on the RADIUS server on page 59) to prevent the 802.1X authentication from being performed after successful MAC authentication or by configuring **mac-authentication dot1x-disable**. In this case, the client is authenticated, and the policies returned by the RADIUS server are applied after successful MAC authentication.

When MAC authentication fails, 802.1X authentication is not attempted, and the configured failure action is applied. However, if the **mac-authentication dot1x-override** command is configured, the clients that failed MAC authentication undergo 802.1X authentication. If 802.1X authentication is successful, the policies returned by the RADIUS server are applied to the port. If 802.1X authentication fails, the failure action is applied to the client.

When the timeout-action is success and the client is dot1x-capable, both authentication methods are tried. The client is placed in the auth-def VLAN, and EAP-SUCCESS is sent by the device. When the timeout-action is critical-vlan and the client is dot1x capable, both authentication methods are tried, the client is placed in the critical VLAN, and EAP-SUCCESS is sent by the device.

FIGURE 7 Authentication sequence: MAC authentication followed by 802.1X authentication



The following list describes how Flexible authentication works in various success, failure, timeout, and dynamic VLAN assignment scenarios:

- When authentication succeeds and RADIUS returns VLAN information, the client is dynamically assigned to the RADIUS assigned VLAN (the MAC address of the client is assigned to the VLAN), and authorization is carried out, depending on the attributes returned from the RADIUS server. For more information, refer to Dynamic VLAN assignment on page 205.
- If RADIUS does not return any VLAN information after authentication, the client is placed in the auth-default VLAN.
- If the authentication fails, failure action is carried out as per the configured failure action, for example, blocking the client or moving the client to the restricted VLAN.

- When the RADIUS server times out and authentication timeout action is configured as "success", the client is authenticated in the auth-default VLAN or the previously authenticated VLAN, depending on the following conditions:
  - If the RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the auth-default VLAN.
  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation.
- When the RADIUS server times out and authentication timeout action is configured as "critical-VLAN", the client is authenticated in the critical VLAN or the previously authenticated VLAN, depending on the following conditions:
  - If the RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the critical VLAN.
  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation.
- When the RADIUS server times out and the authentication timeout action is configured as "failure", the configured auth-failure-actions is performed, for example, moving the client to the restricted VLAN or blocking the client.
- During authentication, when RADIUS returns ACLs and the ACLs are not configured on the ICX device, the client authentication fails by default, resulting in the client being blocked.

Configuration considerations and guidelines for Flexible authentication

- In FastIron release 08.0.90 and later releases, there will no longer be any link flap when a port is being added as a tagged member to a VLAN for the first time or when a port is being removed from the last tagged VLAN. External devices that may have relied on this link flap in the past for any kind of renegotiation must be reconfigured appropriately, or the user must manually flap the interface.
- The RADIUS server must be configured to support a specific authentication method or a combination of authentication methods. A RADIUS server can be configured to use only 802.1X, MAC, or Web authentication or a combination of these authentication methods. For more information about RADIUS configuration, refer to RADIUS Authentication on page 55.
- MACsec and Flexible authentication cannot be configured on the same port.
- Flexible authentication and MVRP cannot be enabled on the same port.
- MVRP dynamic VLANs cannot be configured as authentication VLANs (auth-default, guest, critical, or restricted). This limitation applies at both the global configuration and interface configuration level.
- Flexible authentication cannot be enabled on ports that have any of the following features enabled:
  - Link aggregation
  - Mirroring
  - Static IPv4/IPv6 ACL
  - Selective Q-in-Q
  - VLAN-mapping
  - Route-only
- Incoming traffic on unauthenticated ports is blocked by ICX devices, while allowing for outgoing broadcasts and multicasts to account for waking connected devices that are in a sleep state. This is the default behavior, and there is no configuration option.
- If Web authentication is enabled on the restricted VLAN, critical VLAN, guest VLAN, or a RADIUS assigned VLAN, the device uses Web authentication as a fallback or additional authentication method. Web authentication can be enabled on eight VLANs.

- The client session establishes a relationship between the username and MAC address used for authentication. If attempting to gain access from different clients (with different MAC addresses), the user must be authenticated from each client.
- When a client is denied access to the network, its session is aged out if no traffic is received from the client MAC address over a default hardware aging period (70 seconds), plus a software aging period. Both these age intervals can be changed, or session aging can be disabled altogether. After the denied client session is aged out, traffic from that client is no longer blocked, and the client can be reauthenticated.

802.1X authentication

ICX devices support the IEEE 802.1X standard for authenticating devices attached to LAN ports. Using 802.1X, you can configure an ICX device to grant access to a port based on information supplied by a client to an authentication server.

When a user logs in to a network that uses 802.1X, the ICX 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 provides an alternative to granting network access based on a user IP address, MAC address, or subnetwork.

The ICX implementation of 802.1X supports the following RFCs:

- RFC 2284 PPP Extensible Authentication Protocol (EAP)
- RFC 2865 Remote Authentication Dial In User Service (RADIUS)
- RFC 2869 RADIUS Extensions

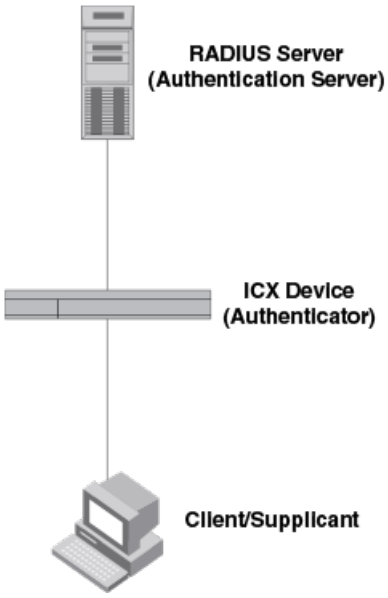
Device roles in an 802.1X configuration

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 8 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 ICX 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 through a hub.

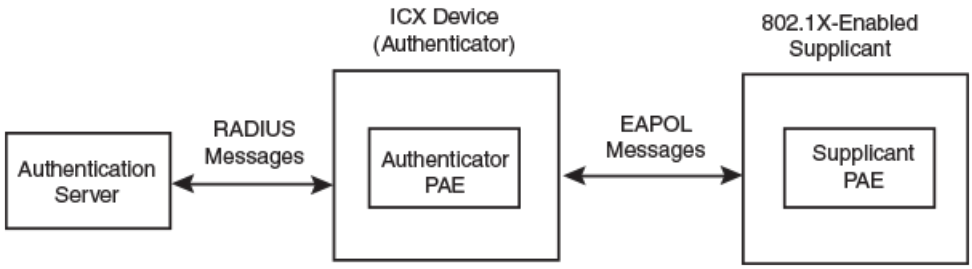
Authentication server: The device that validates the client and specifies whether or not the client may access services on the device. ICX devices support authentication servers running RADIUS.

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 9 Authenticator PAE and supplicant PAE



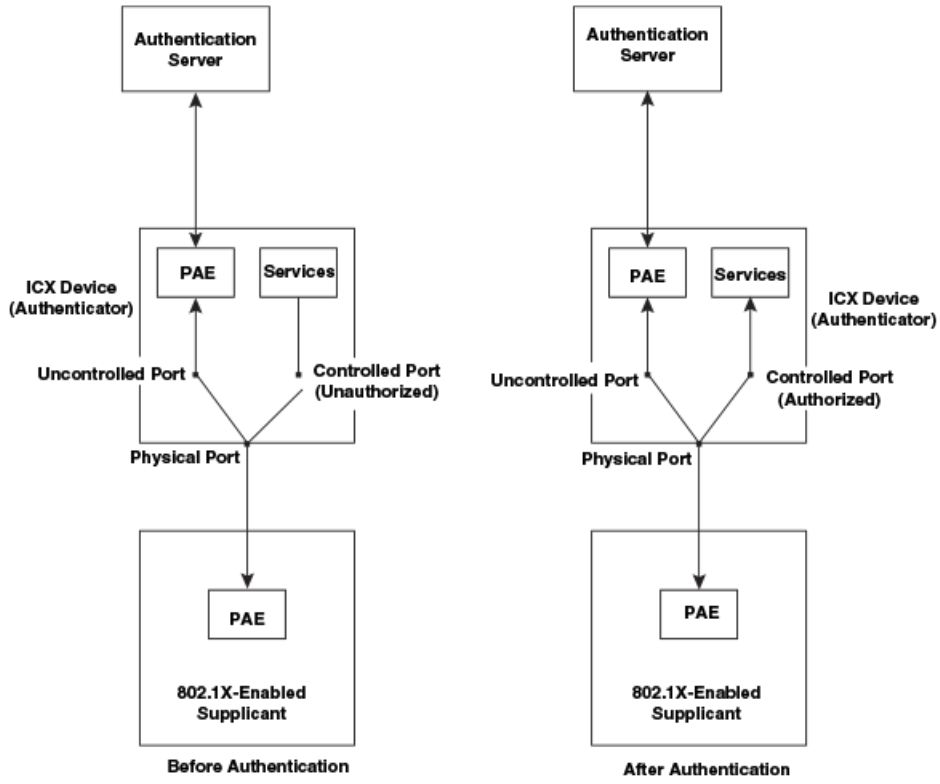
Authenticator PAE: The authenticator PAE communicates with the supplicant PAE, receiving identifying information from the supplicant. Acting as a RADIUS client, the authenticator PAE passes the supplicant information to the authentication server, which decides whether the supplicant can gain access to 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 10 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, the client is authorized to send traffic through the controlled port until the client logs off.

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.

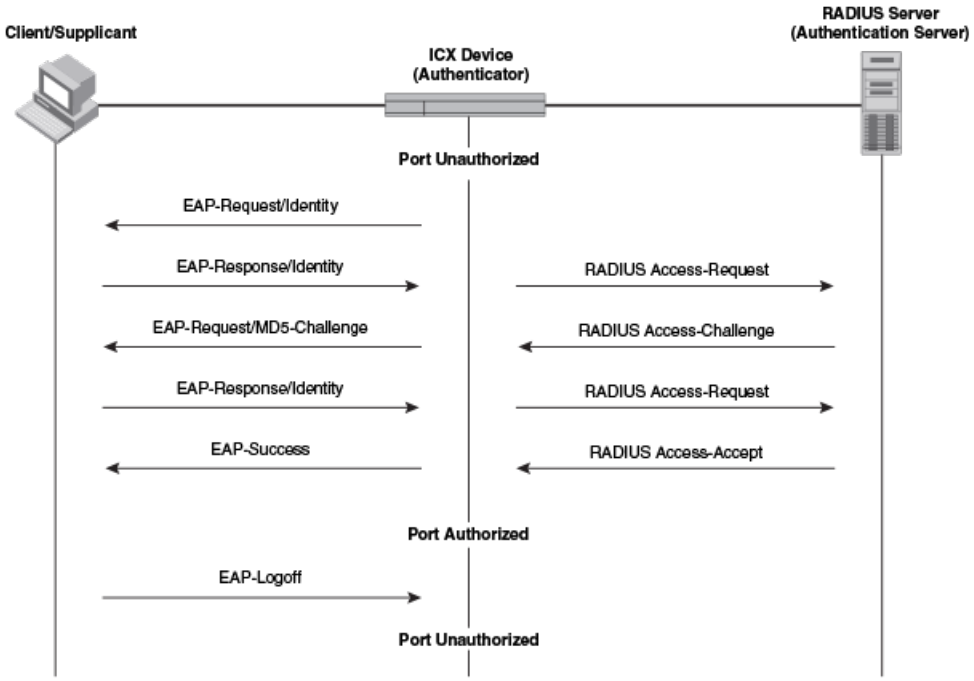
The port control type can be one of the following:

- Force-authorized: The controlled port is placed unconditionally in the authorized state, allowing all traffic. This is the default state for ports on the ICX device.
- Force-unauthorized: The controlled port is placed unconditionally in the unauthorized state.

Message exchange during authentication

The following figure illustrates a sample exchange of messages between an 802.1X-enabled client, an ICX device acting as the authenticator, and a RADIUS server acting as an authentication server.

FIGURE 11 Message exchange between client/supplicant, authenticator, and authentication server



In this example, the authenticator (the ICX device) initiates communication with an 802.1X-enabled client. When the client responds, it is prompted for a username (255 characters maximum) 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 port is authorized. When the client logs off, the port becomes unauthorized again.

The ICX 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, the client port becomes a MAC VLAN member of the specified VLAN. When the client disconnects from the network, the port is removed from the authorized VLAN.

The ICX 802.1X implementation supports dynamically applying an IP ACL to a port in either the ingress or egress direction, based on information received from the authentication server.

If a client does not support 802.1X, authentication cannot take place. The ICX 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 ICX device. When the ICX device does not respond, the client considers the port to be authorized and starts sending normal traffic.

ICX devices support Identity and MD5-challenge requests in EAP Request/Response messages, as well as the following 802.1X authentication challenge types:

- EAP-TLS (RFC 2716): EAP Transport Level Security (TLS) provides strong security by requiring both the client and the authentication server to be identified and validated through the use of public key infrastructure (PKI) digital certificates. EAP-TLS establishes a tunnel between the client and the authentication server to protect messages from unauthorized user eavesdropping activities. Because EAP-TLS requires PKI digital certificates on both the clients and the authentication servers, the roll out, maintenance, and scalability of this authentication method is much more complex than other methods. EAP-TLS is best for installations with existing PKI certificate infrastructures.
- EAP-TTLS (Internet-Draft): The EAP Tunnelled Transport Level Security (TTLS) is an extension of EAP-TLS. Like TLS, EAP-TTLS provides strong authentication measures; however, it requires only the authentication server to be validated by the

client through a certificate exchange between the server and the client. Clients are authenticated by the authentication server using usernames and passwords.

A TLS tunnel can be used to protect EAP messages and existing user credential services such as Active Directory, RADIUS, and LDAP. Backward compatibility for other authentication protocols such as PAP, CHAP, MS-CHAP, and MS-CHAP-V2 are also provided by EAP-TTLS. EAP-TTLS is not considered foolproof and can be fooled into sending identity credentials if TLS tunnels are not used. EAP-TTLS is suited for installations that require strong authentication measures without the use of mutual PKI digital certificates.

- PEAP (Internet-Draft): Protected EAP Protocol (PEAP) is an Internet-Draft that is similar to EAP-TTLS. A PEAP client authenticates directly with the back-end authentication server. The authenticator acts as a pass-through device that does not need to understand the specific EAP authentication protocols.

  Unlike EAP-TTLS, PEAP does not natively support usernames and passwords to authenticate clients against an existing user database such as LDAP. PEAP secures the transmission between the client and the authentication server with a TLS-encrypted tunnel. PEAP also allows other EAP authentication protocols to be used. It relies on the mature TLS keying method for its key creation and exchange. PEAP is best suited for installations that require strong authentication without the use of mutual certificates.

Configuration for these challenge types is the same as for the EAP-MD5 challenge type.

NOTE
If the 802.1X client sends a packet larger than 1500 bytes, you must use the **jumbo** command at the global configuration level of the CLI. If the supplicant or the RADIUS server does not support jumbo frames and if jumbo support is enabled on the switch, you can set the CPU IP MTU size. For more information on setting IP MTU size, refer to the "IP Addressing" chapter in the *Ruckus FastIron Layer 3 Routing Configuration Guide*.

EAP pass-through support

EAP pass-through is supported on ICX devices that have 802.1X enabled. EAP pass-through support is fully compliant with RFC 3748, in which, by default, compliant pass-through authenticator implementations forward EAP challenge request packets of any type.

MAC authentication

MAC authentication is a way to configure an ICX device to forward or block traffic from a client (MAC address) based on authentication information received from a RADIUS server.

MAC authentication is a mechanism by which the ICX device switches or forwards incoming traffic originating from a specific MAC address only if a RADIUS server successfully authenticates the source MAC address. The MAC address itself is used as the username and password for RADIUS authentication. The user does not need to provide a specific username and password to gain access to the network.

The ICX implementation supports dynamic VLAN assignment. If one of the attributes in the Access-Accept message sent by the RADIUS server specifies a VLAN identifier, the client port becomes a MAC VLAN member of the specified VLAN. When the client disconnects from the network, the port is removed from the authorized VLAN.

The ICX implementation also supports dynamically applying an IP ACL to a port in the ingress or egress direction, based on information received from the authentication server.

If the RADIUS server cannot validate the user's MAC address, it is considered an authentication failure, and a specified authentication failure action is applied. The default authentication failure action is to drop traffic from the non-authenticated

MAC address in hardware. You can also configure the device to move the port on which the non-authenticated MAC address was learned into a restricted VLAN.

MAC address formats sent to the RADIUS server

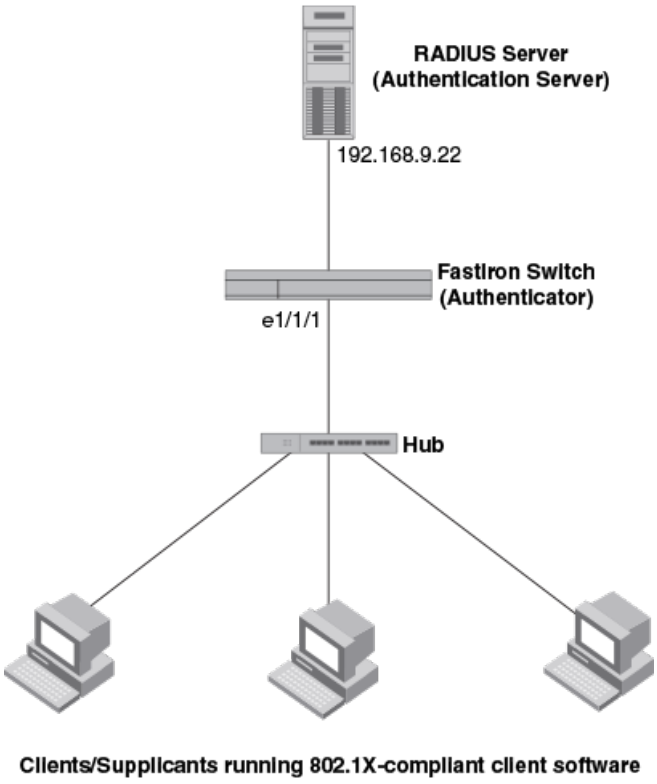
The MAC address of the client is used as the username and password for authentication.

When MAC authentication is configured, the ICX device authenticates the client using the MAC address and the RADIUS server. The device uses the MAC address for both the username and the password in the request sent to the RADIUS server. Several formats can be used to send the MAC address to the RADIUS server, including sending the MAC address in upper case, The format is configurable using the command **mac-authentication password-format**. The lowercase option and xxxxxxxxxxxx are the default format.

Authenticating multiple hosts connected to the same port

ICX devices support 802.1X authentication for ports with more than one host connected to them. The following figure illustrates a sample configuration where multiple hosts are connected to a single port.

FIGURE 12 Multiple hosts connected to a single 802.1X-enabled port



If multiple hosts are connected to a single 802.1X- or or MAC authentication-enabled port, the ICX device authenticates each of them individually. Each host authentication status is independent of the others, so that if one authenticated host disconnects from the network, it has no effect on the authentication status of any of the other authenticated hosts.

By default, traffic from hosts that cannot be authenticated by the RADIUS server is dropped in hardware. As an option, you can configure the ICX device to assign the port to a "restricted" VLAN if authentication of the client is unsuccessful.

How Flexible authentication works for multiple clients

When multiple hosts are connected to a port on an ICX device, Flexible authentication is performed in the following way.

1. One of the 802.1X-enabled clients attempts to log in to a network in which an ICX device serves as an authenticator.
2. The ICX device creates an internal session for the client. The session serves to associate a client MAC address and username with its authentication status. Users trying to gain access from different clients (with different MAC addresses) must be authenticated from each client.
3. The ICX device performs 802.1X or MAC authentication for the client. Messages are exchanged between the ICX device and the client and between the ICX device and the authentication (RADIUS) server. The result of this process is that the client is either successfully authenticated or not authenticated, based on the access policy configured on the RADIUS server.
4. If the client is successfully authenticated, the client session is set to "access-allowed." This means that traffic from the client can be forwarded normally.
5. If authentication for the client is unsuccessful, an authentication failure action is applied. The authentication failure action can be either to drop traffic from the client or to place the client in a restricted VLAN:
   - If the authentication failure action is to drop traffic from the client, the client session is set to "access-denied." This causes traffic from the client to be dropped in hardware.
   - If the authentication failure action is to place the port in a restricted VLAN, the client session is set to "access-restrict." The port is moved to the specified restricted VLAN, and traffic from the client is forwarded normally.
6. The previous steps are repeated for all clients.
7. When the client disconnects from the network, the ICX device deletes the client session. This does not affect the session or authentication status (if any) of the other hosts connected on the port.
8. If any of the clients have IP ACLs sent by the RADIUS server, all such user ACLs are applied to the respective client sessions.

Flexible authentication accounting

When 802.1X or MAC authentication is enabled on the ICX device, RADIUS accounting can also be enabled. This feature enables the ICX device to log information on the RADIUS server about 802.1X- authenticated clients or MAC-authenticated clients. The information logged on the RADIUS server includes the client session ID, MAC address, user name, authenticating physical port, packet statistics, client IP address information, and so on. Flexible authentication accounting works as follows.

1. A RADIUS server successfully authenticates a client.
2. If 802.1X accounting or MAC authentication accounting is enabled, the ICX device sends an 802.1X Accounting Start packet to the RADIUS server, indicating the start of a new session.
3. The RADIUS server acknowledges the Accounting Start packet.
4. The RADIUS server records information about the client.
5. Periodically or whenever a change occurs in the session, such as an IP address update, the ICX device sends an Interim Accounting Stop packet to the RADIUS server.
6. The RADIUS server acknowledges the Interim Accounting Stop packet.
7. When the session is concluded, the ICX device sends an Accounting Stop packet to the RADIUS server, indicating the end of the session.
8. The RADIUS server acknowledges the Accounting Stop packet.

For more information, refer to RADIUS accounting for 802.1X authentication and MAC authentication on page 69.

Change of Authorization (CoA)

Change of authorization (CoA) allows administrators to change authorization dynamically after the device or user is authenticated. As part of authorization, the user or device is given access to specific resources on the network based on the policies or commands downloaded from the RADIUS server. The CoA allows the administrator to change these policies without terminating the sessions. A CoA request packet can be sent by the CoA client (typically a RADIUS or policy server) to change the session authorizations on the ICX device. The request identifies the device and the sessions to be authorized. The following table explains the CoA commands and relevant attributes needed.

NOTE
Only relevant attributes should be present in the CoA request packet. Any additional attributes present in the CoA Request packet sent from the RADIUS client may cause the CoA request to fail (could result in a NAK).

TABLE 20 CoA Commands and RADIUS Attributes

CoA command	Description	RADIUS attributes
Disconnect	Disconnect the specified session	Calling-Station-Id(31), NAS-IP-Address(4)
Modify ACL	Modify ingress/egress ACLs on specified session	Calling-Station-Id(31), NAS-IP-Address(4), Filter-Id(11)
Disable Port	Disable the specified port	Calling-Station-Id(31), NAS-IP-Address(4), Ruckus FlexAuth AVP(20)
Flip Port	Flip the specified port (disable and enable)	Calling-Station-Id(31), NAS-IP-Address(4), Ruckus FlexAuth AVP(20)
Reauth Host	Reauthenticate the specified host (session)	Calling-Station-Id(31), NAS-IP-Address(4), Ruckus FlexAuth AVP(20)

For more information on attributes for RADIUS to support CoA, refer to Company-specific attributes on the RADIUS server on page 59.

Multiple RADIUS servers

Flexible authentication communicates with the RADIUS server to authenticate a new client or reauthenticate an already authenticated client. The ICX device supports multiple RADIUS servers. If communication with one of the RADIUS servers times out, the others are tried in sequential order. If a response from a RADIUS server is not received within a specified time (by default, 3 seconds), the RADIUS request times out, and the device retries the request up to three times. If no response is received, that RADIUS server is marked as down, and the next available RADIUS server is chosen, until all servers are exhausted, or a response is received.

Marking the RADIUS server as down helps in making the authentication process faster, as only the available servers are contacted. When configured, the servers that are down are periodically contacted to check if they are available, and when they become available, they are marked accordingly.

There are several professional and experimental quality RADIUS servers, and all servers are configured with the usernames and passwords of authenticated users. For MAC authentication, the username and password are the MAC address itself. The ICX device uses the MAC address for both the username and the password in the request sent to the RADIUS server. For 8021.X, the username and password are typically configured as unique IDs, which the clients use when they log into the network. For example, given a MAC address of 00:10:94:00:fe:aa, the user's file on the RADIUS server is configured with the username and password both set to 00:10:94:00:fe:aa. If a user using dot1x has to authenticate from the same device, the user profile may have name, password.

Session aging

Session aging is enabled by default, and all sessions are monitored for inactivity so that they can be purged when they are inactive for longer periods of time to conserve resources.

Aging for permitted MAC addresses

Aging for a permitted or non-blocked MAC address occurs in two phases, known as MAC aging and software aging.

The MAC aging time for non-blocked MAC addresses is the length of time specified with the **mac-age** command. The software aging period for non-blocked MAC addresses is configurable, using the **max-sw-age** command (the default is 120 seconds). When the MAC aging period ends, the software aging period begins. When the software aging period ends, the session is aged out.

Aging for denied MAC addresses

Software aging is not applicable for blocked MAC addresses. The hardware aging period for blocked MAC addresses is set to 70 seconds by default, and it can be configured using the **max-hw-age** command. Once the hardware aging period ends, the blocked MAC address ages out so that the session is deleted and can be authenticated again if the ICX device receives traffic from the MAC address.

Disabling MAC address aging

Aging can be disabled for all MAC sessions globally or at the interface level to prevent the MAC sessions from being aged out.

You can disable aging of either the permitted (authenticated and restricted) sessions or the denied sessions. If **disable aging** is configured for permitted MAC sessions, only the permitted sessions are prevented from being aged out, while the denied sessions age out after the hardware aging period. If **disable aging** is configured for denied sessions, only the denied sessions are prevented from being aged out, while the permitted sessions age out based on the MAC aging interval configured using the **mac-age-time** command plus the software aging period.

Periodic reauthentication of authenticated clients

> **NOTE**
> Reauthentication is enabled by default for restricted and critical VLANs. Reauthentication is not supported for guest VLANs.

ICX devices can be configured to periodically reauthenticate the authenticated clients that are connected to interfaces enabled for 802.1X and MAC authentication. When periodic reauthentication is enabled using the **re-authentication** command, the ICX device reauthenticates clients every 3,600 seconds (1 hour) by default. The reauthentication interval is configurable using the **reauth-period** command. The RADIUS server can overwrite this interval for each client using the Session-Timeout and Termination-Action attributes.

> **NOTE**
> With the dead RADIUS server enhancement, RADIUS servers are monitored and marked dead when they don't respond. When no RADIUS servers are available, a client reauthentication attempt would simply result in a timeout. Therefore, when no servers are available, reauthentication is not performed. Instead, the timeout action is performed, and the client continues with the same credentials when the **auth-timeout-action** command configuration is set to success or critical-vlan. This avoids potential issues with some 802.1X clients that are known to go into an inconsistent state after client reauthentication times out and when the **auth-timeout-action** command configuration is set to success.

Denial of Service protection support

A Denial of Service (DoS) attack can occur against the ICX device when a high volume of new source MAC addresses is sent to the device, causing the CPU to be overwhelmed with performing RADIUS authentication for these MAC addresses. In addition, the high CPU usage in such an attack could prevent the RADIUS response from reaching the CPU in time, causing the device to make additional authentication attempts.

You can enable Denial of Service protection using the **authentication dos-protection** command in interface configuration mode. The ICX device does not start forwarding traffic from an authenticated MAC address in hardware until the RADIUS server authenticates the MAC address. Traffic from the non-authenticated MAC addresses is sent to the CPU.

SNMP traps for Flexible authentication

You can enable and disable SNMP traps for Flexible authentication using the **snmp-server enable traps mac-authentication** command. SNMP traps are enabled by default.

Syslog messages for Flexible authentication

Syslog messages are always enabled and sent to the configured log server and to internal logging for significant events during the authentication of clients. Events such as user logged-in, user logged-off, and RADIUS accepted or rejected with list of attributes are logged.

RADIUS attributes for authentication and accounting

Various RADIUS attributes are supported for 802.1Xand MAC authentication and accounting.

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 (RADIUS) server sends the authenticator (the ICX device) a RADIUS Access-Accept message that grants the client access to the network. The Access-Accept message contains attributes set for the user in the user's access profile on the RADIUS server. The user's access profile is used for many functions, such as dynamic VLAN assignment, dynamic IP ACL assignment, session timeout, and authentication order rules for Flexible authentication.

The following table describes the RADIUS attributes that are supported for 802.1X and MAC authentication.

TABLE 21 RADIUS attributes for 802.1X and MAC authentication

Attribute name	Attribute ID	Data Type	Description
Acct-Interim-Interval	85	integer	Indicates the interval for sending the interim updates to the RADIUS server in seconds. The minimum value is 300 seconds.
Calling-Station-Id	31	string	The supplicant MAC address in ASCII format (uppercase only), with octet values separated by a dash (-). For example, 00-00-00-23-19-C0.
Class	25	string	Sent by the server to the client in an Access-Accept. This attribute should not be modified by the

TABLE 21 RADIUS attributes for 802.1X and MAC authentication (continued)

Attribute name	Attribute ID	Data Type	Description
			client; it should be sent to the accounting server when accounting is supported.
Idle-Timeout	28	integer	dle timeout after which the session is cleared when there is no traffic. This is equivalent to configuring the **max-sw-age** command through the CLI.
NAS-Identifier	32	string	Network access server identifier (the hostname of the device).
NAS-IP-Address	4	integer	IP address of the network access server requesting user authentication.
NAS-Port	5	integer	Physical network access server port number that is authenticating the user.
NAS-Port-Id	87	string	Identifier of the network access server port that is authenticating the user.
NAS-Port-Type	61	integer	The port type (physical or virtual) that is authenticating the user.
Service-Type	6	integer	The type of service that is requested by the user or to be provided to the user.
Session-Timeout	27	integer	The maximum number of seconds of service provided to the user before termination of the session or prompt.
Termination-Action	29	integer	The action to be taken by the network access server when the specified service is completed.
Tunnel-Private-Group-ID	81	string	Group identifier for a particular tunneled session.
Tunnel-Medium-Type	65	integer	Indicates which transport medium to use when creating a tunnel for protocols that can operate over multiple transport mediums.
Tunnel-Type	64	integer	Indicates the tunnel protocol that is either in use by a tunnel terminator or to be used by a tunnel initiator.
User-Name	1	string	Indicates the name of the user to be authenticated.

NOTE
Any of the default or configured values on the ICX device are replaced with RADIUS sent attributes, as RADIUS values always take precedence over the values configured on the ICX device.

The following table describes the RADIUS attributes that are supported for 802.1X and MAC accounting.

TABLE 22 RADIUS attributes for 802.1X and MAC accounting

Attribute name	Attribute ID	Data Type	Description
Acct-Authentic	45	Integer	Indicates how the user was authenticated: 1—RADIUS 2—Network access server itself 3—Other remote authentication protocol.
Acct-Delay-Time	41	Integer	Number of seconds that the client has been trying to send this record.
Acct-Input-Octets	42	Integer	Number of octets received from the port while this service is being provided. This attribute can only be present in Accounting-Request records when the Acct-Status-Type is set to stop.
Acct-Input-Packets	47	Integer	Number of packets received from the port while this service is being provided to a framed user. This attribute can only be present in Accounting-Request records when the Acct-Status-Type is set to stop.
Acct-Output-Octets	43	Integer	Number of octets sent to the port while this service is being provided. This attribute can only be present in Accounting-Request records when the Acct-Status-Type is set to stop.
Acct-Output-Packets	48	Integer	Number of packets sent to the port while this service is being provided to a framed user. This attribute can only be present in Accounting- Request records when the Acct- Status-Type is set to stop.
Acct-Session-Id	44	Integer	The account session ID, which is a number from 1 through 4294967295.
Acct-Session-Time	46	Integer	Number of seconds that the user has received service. This attribute can only be present in Accounting- Request records when the Acct- Status-Type is set to stop.
Acct-Status-Type	40	Integer	Indicates whether this Accounting Request marks the beginning (start) or end (stop) of the user service. It also indicates when to send interim updates to the RADIUS server on the status of an active session such as IP address change. The interim update includes the duration of the current session and information on current data usage.

TABLE 22 RADIUS attributes for 802.1X and MAC accounting (continued)

Attribute name	Attribute ID	Data Type	Description
			1—Start 2—Stop 3—Interim Update.
Acct-Terminate-Cause	49	Integer	Specifies the reason for session termination; for example, session timeout, idle timeout, user logoff, admin forced, port down or disabled, system reload, and so on. This attribute is sent out in an Accounting Stop request.
Framed-IPv4-Address	8	IPv4 address	IPv4 address that is assigned on the host or IP routing residential gateway to the interface facing the network access server.
Framed-IPv6-Address	168	IPv6 address	IPv6 address that is assigned on the host or IP routing residential gateway to the interface facing the network access server.
Framed-MTU	12	Integer	Indicates the maximum transmission unit (MTU) to be configured for the user.

Configuring ICX Vendor Specific Attributes on the RADIUS server

If the RADIUS authentication is successful, the RADIUS server sends an Access-Accept message to the ICX device, authenticating the client. The Access-Accept message can include attributes that specify additional information about the client. If MAC authentication and 802.1X authentication are configured on the same port, attributes listed in the following tables can be configured on the RADIUS server.

Attributes should be added to the RADIUS server configuration and configured in the individual or group profiles of the devices tobe authenticated. The ICX device supports two Vendor-IDs: 1991 (Foundry) and 25053 (Ruckus). The Foundry Vendor ID is supported for backward compatibility. For more information, refer to Configuring RADIUS on page 59. The following tables list the attributes for Foundry and Ruckus VSAs.

These attributes are optional.

TABLE 23 Foundry VSAs for RADIUS

Attribute name	Attribute ID	Data type	Description
Foundry-802_1x-enable	6	integer	Specifies whether 802.1X authentication is performed when MAC authentication is successful for a device. This attribute can be set to one of the following: 0 - Do not perform 802.1X authentication on a device that passes MAC authentication.

TABLE 23 Foundry VSAs for RADIUS (continued)

Attribute name	Attribute ID	Data type	Description
			1 - Perform 802.1X authentication when a device passes MAC authentication.
Foundry-802_1x-valid	7	integer	Specifies whether the RADIUS record is valid only for MAC authentication, or for both MAC authentication and 802.1X authentication. This attribute can be set to one of the following: 0 - The RADIUS record is valid only for MAC authentication. Set this attribute to 0 to prevent a user from using their MAC address as the username and password for 802.1X authentication. 1 - The RADIUS record is valid for both MAC and 802.1X authentication.
Foundry-COA-Command-List	10	string	Specifies the CoA command to be performed. This attribute can be set to one of the following: Reauth-host - reauthenticate the host Disable-port – disable the port Flip-port – flip or reset the port.
Foundry-Voice-Phone-Config	11	string	Identifies the client as a voice phone device and optionally specifies the voice phone device configuration. When the device is a phone, LLDP/CDP is configured to voice VLAN to phone, so phone uses a voice session. LLDP supports additional options, such as differentiated services code point (DSCP) and priority, to configure MED policy. The following options can be specified through this VSA: " " –DSCP: 46, priority 5 (default) "dscp:40; priority:4" - DSCP: 40, priority: 4 "dscp:30" - DSCP: 30, priority: 5 "priority:7" - DSCP: 46, priority: 7.

TABLE 24 Ruckus VSAs for RADIUS

Attribute name	Attribute ID	Data type	Description
Ruckus FlexAuth AVP	20	string	The generic name of the attribute is value pair attribute, which can

TABLE 24 Ruckus VSAs for RADIUS (continued)

Attribute name	Attribute ID	Data type	Description
			specify the following attributes (similar to the Foundry VSAs): dot1x-enable (same as Foundry-802.1x-enable) dot1x-valid (same as Foundry-802.1x-valid) coa-attr (same as Foundry-CoA-Command-List) voice-phone (same as Foundry-Voice-Phone-Config).

Support for the RADIUS user-name attribute in Access-Accept messages

ICX devices support the RADIUS user-name (type 1) attribute in the Access-Accept message returned during authentication.

In 802.1X authentication, the user-name attribute is useful when the client does not provide a username in the EAP-response/identity frame and the username is key to providing useful information.

In MAC authentication, the user-name attribute is useful to bind the user-name with the client MAC address, as the client never provides it, and the username is key to providing useful information.

When sFlow forwarding is enabled on a Flexible authentication-enabled interface, the samples taken from the interface include the username string at the inbound or outbound port, or both, if that information is available. For more information on sFlow, refer to the *Ruckus FastIron Monitoring Configuration Guide*.

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 auth** commands, such as **show auth sessions** and **show auth session detail**.

This same information is included as the user-name attribute of RADIUS accounting messages sent to RADIUS accounting servers.

To enable the user-name attribute, add the following information on the RADIUS server.

TABLE 25 RADIUS user-name attribute details

Attribute name	Type	Value
user-name	1	*name* (string)

Dynamic VLAN assignment

After authentication succeeds, a VLAN assignment policy can be applied to control the destination VLAN of the client. Dynamic VLAN assignment allows clients to connect to the network anywhere. Based on their credentials, they are placed in the appropriate VLAN irrespective of the ports to which they are connected.

MAC authentication and 802.1X authentication support dynamic VLAN assignment, where a port can be placed in one or more VLANs based on the VLAN attribute sent from the RADIUS server. With dynamic VLAN assignment, the port becomes a MAC VLAN member of the specified VLAN.

When a client is successfully authenticated, the RADIUS server sends the ICX device a RADIUS Access-Accept message that allows the device to forward traffic from that client (using the MAC address of the client), including the VLAN information using RADIUS attributes to place the client in the designated VLAN. Refer to Configuring the RADIUS server to support dynamic VLAN assignment for authentication on page 206 for a list of the attributes that must be set on the RADIUS server.

A port can be configured with one or multiple authentication methods. If only one authentication is performed, then the VLAN returned from that authentication is used. With multiple methods and based on authentication order, the VLAN from the last authentication is used. If the last authentication does not return any VLAN, the auth-default VLAN is used. This ensures that the user is always placed in a VLAN.

Configuring the RADIUS server to support dynamic VLAN assignment for authentication

Dynamic VLAN assignments from the RADIUS server can be enabled in multiple formats. VLAN assignments can be tagged, untagged, single, multiple, or a combination of tagged and untagged VLANs for different use cases, for example, with client devices such as computers, IP phones, wireless access points, or servers running hypervisors that have multiple Virtual Machines (VMs).

To specify VLAN identifiers on the RADIUS server, add the attributes in the following table to the device (client) profile for MAC authentication. For 802.1X authentication, add these attributes to the user (client) profile.

TABLE 26 Attributes for dynamic VLAN assignment

Attribute name	Type	Value
Tunnel-Type	064	13 (decimal) - VLAN
Tunnel-Medium-Type	065	6 (decimal) - IEEE-802
Tunnel-Private-Group-ID	081	vlan-id or vlan-name (untagged) U:vlan-id of U:vlan-name (untagged) T:vlan-id or T:vlan-name (tagged)

NOTE
Different formats are supported for Tunnel-Private-Group-ID in different FastIron releases. FastIron 08.0.30b and later releases support the following formats:

- vlan-id or vlan-name
- U:vlan-id or U:vlan-name
- T:vlan-id or T:vlan-name
- U:vlan-id T:vlan-name
- multiple T:<>
- U: multiple T:<>

The ICX device interprets the attributes as follows:

- If the Tunnel-Type or the Tunnel-Medium-Type attributes in the Access-Accept message do not have the specified values, the ICX device ignores these Attribute-Value pairs. If the Tunnel-Private-Group-ID is valid, the client is authorized in this VLAN; otherwise, it is authorized in the auth-default VLAN.

- When the ICX device receives and parses the Tunnel-Private-Group-ID attribute, it checks whether the vlan-name string matches the name of a configured VLAN or the vlan-id on the ICX device. If there is a VLAN match, the client port is placed in the VLAN.
- If the vlan-name string does not match either the name or the ID of a VLAN configured on the ICX device, the VLAN name or ID is created and then used.

Authentication success scenarios

The dynamic VLAN assignment depends on the various VLAN formats returned by the RADIUS server.

RADIUS returns only a VLAN identifier or an untagged VLAN identifier

When the Access-Accept message returned by RADIUS contains the VLAN information in either vlan-id or vlan-name or U:vlan-id or U:vlan-name format:

- In single untagged mode (the default), the port membership is removed from the auth-default VLAN and added to the RADIUS-specified VLAN as a MAC VLAN member.

  The following behavior is the default behavior of the device:

  - Subsequent clients that are authenticated with different dynamic VLANs are blocked.
  - If another client is authenticated on the same VLAN, it is permitted in the first client's dynamic VLAN.
  - If another client is authenticated on the port without a RADIUS VLAN, it is permitted in the first client's dynamic VLAN.
  - Once all the clients in the new VLAN age out, the port is moved back to the auth-default VLAN.
- In multiple untagged mode, the port is added as a MAC VLAN member to the RADIUS-specified VLAN without removing its membership from the auth-default VLAN.

RADIUS returns a single or multiple tagged VLAN identifiers

When the Access-Accept message returned by RADIUS contains the VLAN information in either T:vlan-id or T:vlan-id1; T:vlan-id2 format, for MAC authentication, if the authentication is triggered by a tagged packet and if the VLAN matches the tagged VLAN or VLAN list returned by RADIUS, the session is authenticated, and the port becomes a tagged member of all the dynamically assigned VLANs.

RADIUS returns untagged and single or multiple tagged VLAN identifiers

When the Access-Accept message returned by RADIUS contains the VLAN information in U:vlan-id1; T:vlan-id2 or U:vlan-id1; T:vlan-id2; T:vlan-id3; T:vlan-id4 format:

- In single untagged mode (the default), the port membership is removed from the auth-default VLAN and added to the RADIUS- specified VLAN as a MAC VLAN member. It is also added to the tagged VLANs as a tagged member.

  The following behavior is the default behavior of the device:

  - Subsequent clients that are authenticated with different dynamic VLANs are blocked.
  - If another client is authenticated on the same VLAN, it is permitted in the first client's dynamic VLAN.
  - If another client is authenticated on the port without a RADIUS VLAN, it is permitted in the first client's dynamic VLAN.
  - Once all the clients in the new VLAN age out, the port is moved back to the auth-default VLAN.
- In multiple untagged mode, the port is added as a MAC VLAN member to the RADIUS-specified untagged VLAN without removing its membership from the auth-default VLAN. It is also added to the tagged VLANs as a tagged member.

For MAC authentication, if the authentication is triggered by a tagged packet and if the VLAN matches the tagged VLAN or VLAN list returned by RADIUS, the session is authenticated, and the port becomes an untagged member of one VLAN and a tagged member of the other dynamically assigned VLANs.

NOTE
In single untagged mode (the default), if an authenticated client exists on a port and the second client trying to authenticate fails, the port is not moved to a restricted VLAN; instead, the second client is blocked.

Authentication failure scenarios

VLAN assignment in authentication failure cases depends on the authentication mode of the port being single untagged mode (the default) or multiple untagged mode. The authentication failure action configured applies to MAC authentication and 802.1X authentication and can be one of the following:

- Block the client's MAC address.
- Move the client to a restricted VLAN.

A failure action of moving the client to a restricted VLAN works only if the RADIUS server sends ACCESS-REJECT. Any other failure, for example, an undefined ACL or a VLAN movement error, results in blocking the client's MAC Address.

A default ACL with IPv4 or IPv6 filters can also be configured to apply, if the failure action is restricted VLAN.

NOTE
For the default ACL to work, a valid and defined ACL must be configured. Deleting the default ACL may result in unpredictable behavior. Dynamic modification of a default ACL is not supported. If changes must be made to the default ACL, clear all the existing sessions after the modifications are complete.

An authentication failure action of blocking blocks the clients in both single and multiple untagged mode. If the authentication failure action is to move to restricted VLAN, the behavior depends on whether the authentication mode is single untagged mode or multiple untagged mode.

In single untagged mode, the following behaviors apply:

- If the first client's authentication fails, the port membership is moved from the auth-default VLAN to the restricted VLAN.
- If other clients were authenticated previously on the same port, the new client is always blocked. Even after all other clients age out, the new client remains in the VLAN reserved for blocked clients until it ages out.
- If the previous sessions are in a restricted VLAN, the new client is moved to the restricted VLAN.
- If the previous sessions are in the critical VLAN or guest VLAN, the new client is blocked.
- In MAC authentication, if the authentication is initiated by a tagged packet, the client is blocked in the tagged VLAN irrespective of the configured failure action.

In multiple untagged mode, the following behaviors apply:

- If the failure action is configured as a restricted VLAN, the client is moved to the restricted VLAN. If the port is not part of the restricted VLAN, the port is made a MAC-VLAN member of the restricted VLAN.
- For MAC authentication, if the authentication is initiated by a tagged packet, the client is blocked in the tagged VLAN irrespective of the configured failure action.

Authentication server timeout scenarios

VLAN assignment for RADIUS timeout cases depends on whether the authentication mode of the port being single untagged mode (the default) or multiple untagged mode. The authentication timeout action configured applies to MAC authentication and 802.1X authentication and can be one of the following:

- Retry
- Failure
- Success
- Move the client to a critical VLAN

A default ACL with IPv4 or IPv6 filters can also be configured to apply, if the timeout action is critical VLAN or SUCCESS.

NOTE
For the default ACL to work, a valid and defined ACL must be configured. Deleting the default ACL may result in unpredictable behavior. Dynamic modification of a default ACL is not supported. If changes must be made to the default ACL, clear all the existing sessions after the modifications are complete.

Authentication timeout action depends on whether the authentication modes is single untagged mode or multiple untagged mode.

Single untagged mode

- If a RADIUS timeout action is not configured, the MAC session is cleared, and a new authentication is initiated.
- If an authentication timeout action is configured as "failure," the behavior is the same as mentioned in Authentication failure scenarios on page 208.
- If an authentication timeout action is configured as "success," the client is authenticated in the auth-default VLAN or the previously authenticated VLAN, depending on the following conditions:
  - If a RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the auth-default VLAN.
  - If a RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation. The VLAN can be either a dynamic untagged or tagged VLAN.
- If a RADIUS timeout action is configured as "critical-vlan," the action is implemented based on the following conditions:
  - If it is the first client authenticated on the port, the new client is authenticated in the critical VLAN.
  - If the previous sessions are in the auth-default VLAN or RADIUS-assigned VLAN, the new client is blocked.
  - If the previous sessions are in the restricted VLAN or guest VLAN, the MAC address is blocked.
  - If the previous sessions are in the critical VLAN, the client is authenticated in the critical VLAN.
  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation. The VLAN can be either a dynamic untagged or tagged VLAN.

Multiple untagged mode

- If a RADIUS timeout action is not configured, the MAC session is cleared, and a new authentication is initiated.
- If a RADIUS timeout action is configured as "failure," the behavior is the same as mentioned in Authentication failure scenarios on page 208.
- If a RADIUS timeout action is configured as "success," the action is implemented based on the following conditions:
  - If a RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the auth-default VLAN.

  - If a RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation. The VLAN can be either a dynamic untagged or tagged VLAN.
  - For MAC authentication, if the authentication is initiated by a tagged packet, the client is authenticated in the VLAN ID carried by the packet tag value.
- If a RADIUS timeout action is configured as "critical-vlan," the action is implemented based on the following conditions:
  - The client is moved to the critical VLAN.
  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation. The VLAN can be either a dynamic untagged or tagged VLAN.
  - For MAC authentication, if the authentication is initiated by a tagged packet, the client is blocked in the VLAN ID carried by the packet tag value.

Authentication client timeout scenarios (no response to EAP packets)

The dynamic VLAN assignment when the client does not respond to EAP packets is applicable only to 802.1X authentication. VLAN assignment when the client does not respond to the EAP packets depends on whether the authentication mode of the port is single untagged mode (the default) or multiple untagged mode.

Single untagged mode

- If there is no response from the client for EAP packets and if the guest VLAN is not configured, the behavior is the same as mentioned in Authentication failure scenarios on page 208.
- If the guest VLAN is configured:
  - If it is the first client on the port, the client is authenticated in the guest VLAN.
  - If the previous sessions are in a different RADIUS-assigned VLAN, the client is blocked.
  - If the previous sessions are in the guest VLAN, the new client is permitted in the guest VLAN.
  - If the previous sessions are in a critical VLAN or restricted VLAN, the client is blocked.
  - If the previous sessions are in the guest VLAN and the new client is dot1x-capable, then existing guest MAC sessions are cleared before permitting the new client in the auth-default VLAN or RADIUS-assigned VLAN.

Multiple untagged mode

If there is no response from the client for EAP packets and if the guest VLAN is configured, the port is moved to the guest VLAN; otherwise, the failure action is carried out.

Automatic removal of dynamic VLAN assignments for 802.1X and MAC authenticated ports

By default, the ICX device removes any association between a port and a dynamically assigned VLAN when authenticated MAC sessions for that tagged or untagged VLAN have expired on the port. Thus, RADIUS-specified VLAN assignments are not saved to the ICX `running-config` file. When the **show running-config** command is issued during a session, dynamically assigned VLANs are not displayed, although they can be displayed with the **show vlan** and **show authentication sessions** commands.

Dynamic ACLs in authentication

After successful authentication, different network policies can be applied to restrict the way the client accesses network resources. The 802.1X authentication and MAC authentication implementations support dynamically applying IPv4 ACLs and IPv6 ACLs to a port, based on information received from an authentication server.

When a client or supplicant is authenticated, the authentication server (the RADIUS server) sends the authenticator (the Ruckus ICX 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 profile for 802.1X authentication or the device profile for MAC authentication on the RADIUS server.

If the Access-Accept message contains the Filter-Id (type 11), the Ruckus ICX device can use information in the attribute to apply an IP ACL to the authenticated port. This IP ACL filter applies to the port for as long as the client is connected to the network. The IP ACL is removed from the corresponding port when the client logs out, the port goes down, or the MAC session ages out.

The Ruckus ICX device uses information in the Filter-Id as follows:

- The Filter-Id attribute can specify the number of an existing IPv4 ACL or IPv6 ACL configured on the Ruckus ICX device.
- The attribute can specify the name of an existing IPv4 ACL or IPv6 ACL configured on the ICX device.

Configuration guidelines for dynamic ACLs

The following restrictions apply to dynamic IPv4 ACLs and IPv6 ACLs:

- The name in the Filter-Id attribute is case-sensitive.
- When an ACL is created, make sure to immediately add at least one rule to the ACL.
- IPv6 ACL is not supported for Web authentication.
- Both dynamically assigned IPv4 ACLs and IPv6 ACLs can be applied together.
- Both inbound and outbound dynamic IPv4 ACLs and IPv6 ACLs are supported. Only one ACL for each type can be sent from the RADIUS server.
- If both IPv4 ACLs and IPv6 ACLs are applied, only one inbound type or one outbound type of each filter is allowed.
- A maximum of one IP ACL per client can be configured for IPv4 inbound, IPv4 outbound, IPv6 inbound, and IPv6 outbound on an interface.
- Static ACLs are not supported on the 802.1X authentication-enabled or the MAC authentication-enabled port.
- Concurrent operation of dynamic IP ACLs and static IP ACLs is not supported.
- Dynamic IP ACL assignment with 802.1X is not supported in conjunction with any of the following features:
  - Protection against ICMP or TCP Denial of Service (DoS) attacks
  - Policy-based routing
  - Web authentication ACL
- Deletion of an ACL ID is not recommended when ACLs are used by Flexible authentication sessions or Web authentication sessions.
- Dynamic ACLs support either one IPv4 address or up to four IPv6 addresses.
- Changing an ACL ID in the RADIUS user profile is not supported during reauthentication.

Dynamically applying existing ACLs

When a port is authenticated, an IPv4 ACL or IPv6 ACL that exists in the running-config file on the Ruckus ICX device can be dynamically applied to the port. To do this, you configure the Filter-Id (type 11) attribute on the RADIUS server.

The syntax in the following table is used to configure the Filter-Id attribute to refer to an IP ACL.

TABLE 27 Syntax for configuring the Filter-Id attribute

Syntax	Description
ip.*number***.in**, **ip.***name***.in**	Applies the specified numbered or named IPv4 ACL to the authenticated port in the inbound direction.
ip.*number***.out**, **ip.***name***.out**	Applies the specified numbered or named IPv4 ACL to the authenticated port in the outbound direction.
ip6.*number***.in**, **ip6.***name***.in**	Applies the specified numbered or named IPv6 ACL to the authenticated port in the inbound direction.
ip6.*number***.out**, **ip6.***name***.out**	Applies the specified numbered or named IPv6 ACL to the authenticated port in the outbound direction.

The following table lists examples of values that you can assign to the Filter-Id attribute on the RADIUS server to refer to IP ACLs configured on a Ruckus ICX device.

TABLE 28 Sample Filter-Id attribute values on the RADIUS server

Possible values for the Filter-Id attribute on the RADIUS server	ACL configured on the Ruckus device
ip.102.in	access-list 102 deny ip any 10.1.0.0 0.0.0.255 access-list 102 permit ip any any
ip.fdry_filter.in	ip access-list extended fdry_filter deny ip any 10.1.0.0 0.0.0.255 permit ip any any

Support for IP Source Guard protection

The ICX proprietary Source Guard Protection feature, a form of IP Source Guard, can be used in conjunction with Flexible authentication.

When IP Source Guard Protection is enabled using the **authentication source-guard-protection enable** command in interface configuration mode, IP traffic is blocked until the system learns the IP address. Once the IP address is validated, traffic containing that source IP address is permitted.

NOTE
In Flexible authentication, IP Source Guard Protection is applicable only for IPv4 traffic.

When a Flexible authentication session is created on a port that has IP Source Guard Protection enabled, the session either applies a dynamically created IP Source Guard ACL entry or uses the dynamic IP ACL assigned by the RADIUS server. If a dynamic IP ACL is not assigned, the session uses the IP Source Guard ACL entry. The IP Source Guard ACL entry can be **permit ip secure-ip any**, where **secure- ip** is obtained from the ARP Inspection table or from the DHCP Secure table. The DHCP Secure table includes DHCP Snooping and Static ARP Inspection entries.

The IP Source Guard ACL entry is not written to the running-config file. However, you can view the configuration using the **show authentication sessions** command.

NOTE
The secure MAC-to-IP mapping is assigned at the time of authentication and remains in effect as long as the session is active. The existing session is not affected if the DHCP Secure table is updated after the session is authenticated and while the session is still active. Change of IP address is supported.

The IP Source Guard ACL permit entry is removed when the session expires or is cleared.

For more information about IP Source Guard, refer to *IP Source Guard* in the *DHCPv4* chapter of the *Ruckus FastIron DHCP Configuration Guide*.

Configuring Flexible authentication

Flexible authentication requires some prerequisite tasks that must be performed before executing Flexible authentication configurations at the global and interface levels. Flexible authentication configurations also include 802.1X authentication-specific and MAC authentication-specific configurations.

Flexible authentication configuration prerequisites

Before you configure Flexible authentication, you must establish communication between the devices and the authentication server. The following items cover the configuration steps that are required before you configure Flexible authentication:

- Configure the ICX device interaction with the authentication server by configuring an authentication method list for 802.1X and specifying RADIUS as an authentication method. The method list takes care of 802.1X and MAC authentication. For more information, refer to AAA operations for RADIUS on page 57.

```
device(config)# aaa authentication dot1x default radius
```

- Configure the RADIUS server to authenticate access to the Ruckus ICX device. For more information, refer to AAA operations for RADIUS on page 57.

```
device(config)# radius-server host 10.20.64.208 auth-port 1812 acct-port 1813 default key secretkey
dot1x mac-auth
```

- After successful authentication, the client is moved to the RADIUS-assigned VLAN. Configure a VLAN as the auth-default VLAN to enable authentication. When any port is enabled for 802.1X or MAC authentication, the port is moved into this VLAN by default. Specific VLANs (for example, guest VLAN, restricted VLAN, and critical VLAN) can be configured to place the clients in various VLANs based on authentication failure and timeout scenarios.

```
device(config)# vlan 20 name auth-default-vlan
```

- After a successful authentication, user access can be limited by ACLs. ACLs must be preconfigured on the ICX device, and the RADIUS server can return the ACL ID or name. If the ACL matches the ACL configured on the device, it is applied to the port.

```
device(config)# access-list 100 permit ip any any
```

  NOTE
  The source IP must be either be the user's IP address or “any” because the Ruckus ICX device dynamically learns the IP addresses of the clients (source). The destination network is user configurable.

  For more information on ACL configuration, refer to IPv4 ACLs on page 106. For more information about dynamic ACL assignment, refer to Dynamic ACLs in authentication on page 211.
- If any of the clients need to be statically authenticated or denied access, the MAC addresses of such clients can be configured through MAC filters and applied on authentication-enabled ports as authentication filters.

```
device(config)# mac filter 1 permit/deny xxxx.xxxx.xxxx FFFF.FFFF.FFFF any
```

Configuring Flexible authentication globally

The following steps configure Flexible authentication at the global level.

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

```
device# configure terminal
```

2. Enter the **authentication** command to enter authentication configuration mode.

```
device(config)# authentication
```

   All the global authentication configurations are available in the authentication configuration mode.

3. (Optional) Enter the **auth-order mac-auth dot1x** command to change the sequence of authentication methods to MAC authentication followed by 802.1X authentication if required.

   NOTE
   If the 802.1X authentication and MAC authentication methods are enabled on the same port, by default, the authentication sequence is set to perform 802.1X authentication followed by MAC authentication.

```
device(config-authen)# auth-order mac-auth dot1x
```

4. Enter the **auth-default-vlan** command to configure the authentication default VLAN (auth-default VLAN).

   NOTE
   The auth-default VLAN must be configured to enable Flexible authentication before enabling 802.1X authentication or MAC authentication.

```
device(config-authen)# auth-default-vlan 2
```

   All ports are moved to the auth-default VLAN as MAC VLAN member when 802.1X authentication or MAC authentication is enabled. The client remains in the auth-default VLAN if the RADIUS server does not return VLAN information upon authentication or if the RADIUS timeout action is specified as "success" when RADIUS server is not reachable.

5. (Optional) Enter the **restricted-vlan** command to configure the restricted VLAN.

```
device(config-authen)# restricted-vlan 4
```

   When a restricted VLAN is configured, you can configure the authentication failure action as moving the client to the restricted VLAN. If a restricted VLAN is not configured, when authentication fails, the client's MAC address is blocked in the hardware.

6. (Optional) Enter the **auth-fail-action** command to move the port to the restricted VLAN after authentication failure.

```
device(config-authen)# auth-fail-action restricted-vlan
```

7. (Optional) Enter the **critical-vlan** command to configure the VLAN in which the port should be placed when the RADIUS server times out while authenticating or reauthenticating.

```
device(config-authen)# critical-vlan 20
```

8. (Optional) Enter the **auth-timeout-action** command to move the port to the critical VLAN after RADIUS authentication timeout.

```
device(config-authen)# auth-timeout-action critical-vlan
```

9. (Optional) Enter the **auth-mode** command to enable multiple untagged mode, which allows Flexible authentication-enabled ports to be members of multiple untagged VLANs or single-host and multiple-hosts mode.

   NOTE
   By default, a Flexible authentication-enabled port can be a member of only one untagged VLAN (single-untagged mode), and other clients that are authenticated with different dynamic untagged VLANs are blocked.

```
device(config-authen)# auth-mode multiple-untagged
```

10. (Optional) Enter the **disable-aging permitted-mac-only** command to prevent the permitted MAC sessions from being aged out. Or enter the **disable-aging denied-mac-only** to prevent the denied MAC sessions from aging out.

```
device(config-authen)# disable-aging permitted-mac-only
```

    NOTE
    You can disable aging of either the permitted (authenticated and restricted) sessions or the denied sessions. Once configured, MAC addresses that are authenticated or denied by a RADIUS server are not aged out if no traffic is received from the MAC address for a certain period of time. Aging for a permitted or non-blocked MAC address occurs in two phases, MAC aging and software aging. The MAC aging interval is configured using the **mac-age-time** command. By default, **mac-age-time** is set to 300 seconds. After the normal MAC aging period for permitted clients (or clients in a restricted VLAN), the software aging period begins. The **max-sw-age** command is used to specify the software aging period and by default is set to 120 seconds. After the software aging period ends, the client session ages out and is removed from the session table.
    If during software aging, traffic is received from a client, the MAC address of the client is updated in the hardware table, and the client continues to communicate. Software aging is not applicable for blocked MAC addresses.
    The hardware aging period for blocked MAC addresses is set to 70 seconds by default, and it can be configured using the **max-hw-age** command. Once the hardware aging period ends, the blocked MAC address ages out and can be authenticated again if the ICX device receives traffic from the MAC address.

11. (Optional) Enter the **max-hw-age** command to configure the hardware aging period for denied MAC addresses.

```
device(config-authen)# max-hw-age 160
```

12. (Optional) Enter the **max-sw-age** command to configure the software aging period.

```
device(config-authen)# max-sw-age 160
```

13. (Optional) Enter the max-sessions command to configure the number of clients allowed on a port, the default being 2.

```
device(config-authen)# max-sessions 32
```

14. (Optional) Enter the **re-authentication** command to configure the ICX device to periodically reauthenticate the authenticated clients.

```
device(config-authen)# re-authentication
```

    NOTE
    When the periodic reauthentication is enabled, the device reauthenticates clients every 3600 seconds (one hour) by default. The reauthentication interval configured using the **reauth-period** command takes precedence.

15. (Optional) Enter the **reauth-period** command to configure the interval at which authenticated clients are reauthenticated. The default period is an hour, or 3600 seconds.

```
device(config-authen)# reauth-period 2000
```

16. (Optional) Enter the **reauth-timeout** command to configure the interval at which non-authenticated clients in restricted or critical or guest access are reauthenticated. The default period is 300 seconds.

NOTE
Reauthentication is supported for restricted and critical access and not supported for guest access.

```
device(config-authen)# reauth-timeout 120
```

17. (Optional) Specify the IPv4 or IPv6 ingress or egress ACLs to be applied when clients are non-authenticated for various reasons, such as auth-failure, auth-timeout, and access becomes restricted, critical, or guest. Authenticated clients can be assigned ACLs by the RADIUS server.

```
device(config-authen)# default-acl ipv4/ipv6 <acl> in/out
```

18. (Optional) Specify the voice VLAN to be used to add the port as tagged in voice VLAN when it is not provided by the RADIUS server and when the clients are non-authenticated for various reasons, such as auth-failure and auth-timeout.

```
device(config-authen)# voice-vlan 200
```

Configuring Flexible authentication on an interface

The following steps configure Flexible authentication at the interface level.

NOTE
Configuration at the interface level overrides related configuration at the global level. The global configuration is still applicable to other ports that do not have a per-port configuration. Refer to Configuring Flexible authentication globally on page 214 for more information.

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command to enter interface configuration mode.

```
device(config)# interface ethernet 1/1/1
```

3. (Optional) Enter the **authentication auth-order mac-auth dot1x** command to change the sequence of authentication to MAC authentication followed by 802.1X authentication if required.

```
device(config-if-e1000-1/1/1)# authentication auth-order mac-auth dot1x
```

4. Enter the **authentication auth-default-vlan** command to configure the authentication default VLAN (auth-default VLAN).

```
device(config-if-e1000-1/1/1)# authentication auth-default-vlan 30
```

5. (Optional) Enter the **authentication auth-mode** command to enable the multiple untagged mode on a specific Flexible authentication-enabled port and allow it to be a member of multiple untagged VLANs.

```
device(config-if-e1000-1/1/1)# authentication auth-mode multiple-untagged
```

6. (Optional) Enter the **authentication disable-aging permitted-mac-only** or the **authentication disable-aging denied-mac-only** command to prevent the permitted or denied MAC sessions from being aged out from a port.

```
device(config-if-e1000-1/1/1)# authentication disable-aging permitted-mac-only
```

7. (Optional) Enter the **authentication max-sessions** command to specify the maximum limit of authenticated MAC sessions on an interface.

```
device(config-if-e1000-1/1/1)# authentication max-sessions 32
```

8. (Optional) Enter the **authentication dos-protection** command to enable Denial of Service (DoS) authentication protection on an interface.

```
device(config-if-e1000-1/1/1)# authentication dos-protection mac-limit 256
```

   NOTE
   You can also configure the Ruckus ICX device to limit the rate of authentication attempts sent to the RADIUS server.

9. (Optional) Enter the **authentication source-guard-protection** command to enable IP Source Guard Protection along with authentication on an interface.

```
device(config-if-e1000-1/1/1)# authentication source-guard-protection enable
```

10. (Optional) Specify the voice VLAN to be used to add the port as tagged in the voice VLAN when it is not provided by the RADIUS server and when the clients are non-authenticated for various reasons, such as auth-failure and auth-timeout.

```
device(config-if-e1000-1/1/1)# authentication voice-vlan 300
```

11. (Optional) Allow tagged packet processing when the port is not tagged, which may be the case when multiple VMs are connected to the port so that they can be authenticated with MAC authentication, and automatic tagging of the port helps. This option is disabled by default.

```
device(config-if-e1000-1/1/1)# authentication allow-tagged
```

12. (Optional) Enter the auth-filter command to apply the specified filter on the interface so that the MAC addresses defined in the filter (MAC filter) need not go through authentication.

```
device(config-if-e1000-1/1/1)# authentication auth-filter 2 4
```

    The source MAC addresses defined using the **mac-filter** command are considered pre-authenticated and are not subject to authentication. A client can be authenticated in an untagged VLAN or tagged VLAN using the MAC address filter. If the authentication filter has a tagged VLAN configuration, the clients are authenticated in the auth-default VLAN and the tagged VLAN provided in the auth-filter. The clients authorized in the auth-default VLAN allow both untagged and tagged traffic. The auth-filter is defined using the **mac-filter** command.

Enabling 802.1X authentication

The following steps are for enabling and activating 802.1X authentication and for configuring certain 802.1X-specific commands.

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

```
device# configure terminal
```

2. Enter the **authentication** command to enter authentication mode.

```
device(config)# authentication
```

3. Enter the **dot1x enable** command to enable 802.1X authentication.

```
device(config-authen)# dot1x enable
```

4. Enter the **dot1x enable** { **all** | **ethernet** *unit/slot/pot* [**to** *unit/slot/pot*] } command to enable 802.1X authentication on all interfaces, a single interface, or a specific set of interfaces.

```
device(config-authen)# dot1x enable all
```

5. 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.

   The following example configures the command globally.

```
device(config-authen)# dot1x port-control auto all
```

   The following example configures the command on a single interface. (Interface configuration overrides global configuration if they differ.)

```
device(config-authen)# device(config-authen)# dot1x port-control auto ethernet 1/1/1
```

   Once the client passes authentication, the port becomes authorized. This activates authentication on an 802.1X-enabled interface. The controlled port remains in the authorized state until the client logs off.

6. (Optional) Enter the **dot1x guest-vlan** command to configure the VLAN into which the port should be placed when the client's response to the dot1x requests for authentication times out.

```
device(config-authen)# dot1x guest-vlan
```

7. (Optional) Configure the timeout parameters that determine the time interval for client reauthentication and EAP retransmissions using the following commands:
   - Enter the **dot1x timeout quiet-period** command to configure the amount of time the ICX device should wait before reauthenticating the client.

```
device(config-authen)# dot1x timeout quiet-period 30
```

   - Enter the **dot1x timeout tx-period** command to configure the amount of time the ICX device should wait before retransmitting EAP-Request/Identity frames to the client.

```
device(config-authen)# dot1x timeout tx-period 30
```

   - Enter the **dot1x timeout supplicant** command to configure the amount of time the ICX device should wait before retransmitting RADIUS EAP-Request/Challenge frames to the client.

```
device(config-authen)# dot1x timeout supplicant 30
```

   Based on the timeout parameters, the client is reauthenticated, and EAP-Request/Identity frames and EAP-Request/Challenge frames are retransmitted.

8. (Optional) Enter the **dot1x max-reauth-req** command to configure the maximum number of times EAP-Request/Identity frames are sent for reauthentication after the first authentication attempt.

```
device(config-authen)# dot1x max-reauth-req 4
```

   If no EAP Response/Identity frame is received from the client after the specified number of EAP-Request/Identity frame retransmissions, the device restarts the authentication process with the client.

9. (Optional) Enter the **dot1x max-req** command to configure the maximum number of times EAP-Request/Challenge frames are retransmitted when an EAP Response/Identity frame is not received from the client.

```
device(config-authen)# dot1x max-req 3
```

10. (Optional) Enter the **dot1x macuath-override** command to configure the device to perform MAC authentication after 8021.x authentication, if 802.1x authentication fails for the clients.

NOTE
This command is applicable only when the authentication sequence is configured as 8021.x authentication followed by MAC authentication.

```
device(config-authen)# dot1x macauth-override
```

Enabling MAC authentication

The following steps enable MAC authentication and include certain Flexible authentication configurations specific to MAC authentication.

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

```
device# configure terminal
```

2. Enter the **authentication** command to enter authentication mode.

```
device(config)# authentication
```

3. Enter the **mac-authentication enable** command to enable MAC authentication.

```
device(config-authen)# mac-authentication enable
```

4. Enter the **mac-authentication enable** { **all** | **ethernet** *stack/slot/pot* } command to enable MAC authentication on all interfaces or a specific interface.

```
device(config-authen)# mac-authentication enable all
```

5. (Optional) Enter the **mac-authentication password-format** command to configure the format in which the MAC address is sent to the RADIUS server for authentication.

   By default, the MAC address is sent to the RADIUS server in the xxxxxxxxxxxx format in lowercase. As an option, you can change the address to uppercase. You can specify one of the following formats:

   - xx-xx-xx-xx-xx-xx
   - xx:xx:xx:xx:xx:xx
   - xxxx.xxxx.xxxx
   - xxxxxxxxxxxx

```
device(config-authen)# mac-authentication password-format xx-xx-xx-xx-xx-xx upper-case
```

6. (Optional) Enter the **mac-authentication password-override** command to specify a user-defined password instead of the MAC address for MAC authentication.

NOTE
The password can contain up to 32 alphanumeric characters but must not include blank spaces.

```
device(config-authen)# mac-authentication password-override ts54fs
```

7. (Optional) Enter the **mac-authentication dot1x-disable** command to configure the device not to perform 802.1X authentication after MAC authentication when MAC authentication succeeds for the client. This is enabled by default, unless overruled by the RADIUS server through a dot1x-enable attribute.

   NOTE
   This command is applicable only when the authentication sequence is configured as MAC authentication followed by 802.1X authentication.

```
device(config-authen)# mac-authentication dot1x-disable
```

8. (Optional) Enter the mac-authentication dot1x-override command to configure the device to perform802.1X authentication after MAC authentication, if MAC authentication fails for the client.

   NOTE
   This command is applicable only when the authentication sequence is configured as MAC authentication followed by 802.1X authentication.

```
device(config-authen)# mac-authentication dot1x-override
```

Excluding the RADIUS server for login features

You can specify whether the RADIUS server can be used for login features such as Telnet, SSH, console, EXEC, or Web-management AAA. The following task excludes the RADIUS server for all login features.

1. Use the **configure terminal** command to enter global configuration mode.

```
device# configure terminal
```

2. Use the **radius-server host** command, specifying an IP address, and using the **auth-port** *port-num*, **acct-port** *port-num*, **default**, **key** *key-string*, and **no-login** parameters.

```
device(config)# radius-server host 10.26.67.13 auth-port 1812 acct-port 1813 default key ruckus no-
login
```

   The example configures default UDP ports for authorization and accounting. The RADIUS key for the server is configured as “ruckus." The **no-login** keyword is configured so that this RADIUS server cannot be used for Flexible authentication, Telnet, SSH, console, EXEC, or Web-management AAA.

The following example excludes the RADIUS server for login and flexible authentication features.

```
device# configure terminal
device(config)# radius-server host 10.26.67.13 auth-port 1812 acct-port 1813 default key ruckus no-login
```

The following example configures the RADIUS server to be used for login features and Flexible authentication.

```
device# configure terminal
device(config)# radius-server host 10.26.67.13 auth-port 1812 acct-port 1813 default key ruckus
```

The following example configures the RADIUS server to be used for both MAC authentication and login features.

```
device# configure terminal
device(config)# radius-server host 10.26.67.13 auth-port 1812 acct-port 1813 default key ruckus mac-auth
```

The following example uses the RADIUS server for Flexible authentication modules.

```
device# configure terminal
device(config)# radius-server host 10.26.67.13 auth-port 1812 acct-port 1813 default key ruckus macauth
dot1x no-login
```

Displaying authentication information

You can display the following authentication information:

- The authentication configuration on the device and on individual ports
- Statistics about the EAPOL frames passing through the device
- Hosts or user session information
- Authentication-enabled ports dynamically assigned to a VLAN
- User-defined and dynamically applied IP ACLs currently active on the device.

Displaying configuration

The output of the **show authentication configuration** command indicates the global configuration details for authentication as shown in the following example.

```
device# show auth configuration
Auth:
Auth Order                          : mac-auth dot1x
Default VLAN                        : 10
Default Voice VLAN                  : Not configured
Auth Mode                           : Multiple Untagged Mode
Restricted VLAN                     : Not configured
Critical VLAN                       : Not configured
Auth Failure Action                 : Block traffic
Auth Timeout Action                 : Treat as a successful authentication
MAC Session Aging                   : Enabled
Re-authentication                   : Enabled
Reauth-period                       : 180 seconds
Reauth-timeout                      : 120 seconds
Session Max SW-Age                  : 120 seconds
Session Aax HW-Age                  : 70 seconds
Max Sessions                        : 2
MAC-Auth:
Status                              : Enabled
802.1X Override                     : Enabled
Password Override                   : Disabled
Password Format                     : xxxx.xxxx.xxxx
802.1X:
Status                              : Enabled
Protocol Version                    : 1
PAE Capability                      : Authenticator Only
MAC-Auth Override                   : Disabled
Guest VLAN                          : Not configured
Quiet-period                        : 60 seconds
TX-period                           : 30 seconds
Supplicant-timeout                  : 30 seconds
Max Reauth Requests                 : 2
Max Frame Retries                   : 2
```

To display the configuration details for authentication on an interface, enter the **show auth configuration ethernet** command followed by the port number as shown in the following example.

```
device# show auth configuration ethernet 1/1/1
Port 1/1/1 Configuration:
Auth Order                          : mac-auth dot1x
Auth Mode                           : Multiple Untagged Mode
Auth Failure Action                 : Block traffic
Auth Timeout Action                 : Treat as a successful authentication
DoS Protection                      : Disabled (limit = 512)
Source-guard Protection             : Disabled
Aging                               : Enabled
Max Sessions                        : 32
```

```
Reauth-timeout                    : 120 seconds
802.1X Port-Control               : Auto
```

Displaying statistics

To display statistics for an individual port, enter the **show authentication statistics** command as shown in the following example.

```
device# show authentication statistics ethernet 1/1/1
Port 1/1/1 Statistics:
802.1X:
RX EAPOL Start:                   0
RX EAPOL Logoff:                  0
RX EAPOL Invalid:                 0
RX EAPOL Total:                   0
RX EAP Resp/Id:                   0
RX EAP Resp other than Resp/Id:   0
RX EAP Length Error:              0
Last EAPOL Version:               0
Last EAPOL Source:                0000.0050.0B83
TX EAPOL Total:                   217
TX EAP Req/Id:                    163
TX EAP Req other than Req/Id:     0
MAC-Auth:
Accepted Sessions:                0
Rejected Sessions:                0
Inprogress Sessions:              0
Attempted Sessions:               0
Number of Errors:                 0
```

The following table describes the information displayed for an interface by the **show authentication statistics** command.

TABLE 29 Output from the show authentication statistics command

Field	Statistics
RX EAPOL Start	The number of EAPOL-Start frames received on the port
RX EAPOL Logoff	The number of EAPOL-Logoff frames received on the port
RX EAPOL Invalid	The number of invalid EAPOL frames received on the port
RX EAPOL Total	The total number of EAPOL frames received on the port
RX EAP Resp/Id	The number of EAP-Response/Identity frames received on the port
RX EAP Resp other than Resp/Id	The total number of EAPOL-Response frames received on the port that were not EAP-Response/Identity frames
RX EAP Length Error	The number of EAPOL frames received on the port that have an invalid packet body length
Last EAPOL Version	The version number of the last EAPOL frame received on the port
Last EAPOL Source	The source MAC address in the last EAPOL frame received on the port
TX EAPOL Total	The total number of EAPOL frames transmitted on the port
TX EAP Req/Id	The number of EAP-Request/Identity frames transmitted on the port
TX EAP Req other than Req/Id	The number of EAP-Request frames transmitted on the port that were not EAP-Request/Identity frames
Accepted Sessions	Number of MAC authentication sessions accepted
Rejected Sessions	Number of MAC authentication sessions rejected
In Progress Sessions	Number of MAC authentication sessions which are in progress
Attempted Sessions	Number of MAC authentication sessions attempted
Number of Errors	Number of errors encountered while processing sessions

Displaying the authentication sessions

Use the **show authentication sessions** command to view details of the 802.1X or MAC-authentication sessions, such as the ports, MAC addresses, IP addresses, and VLANs, as shown in the following example.

```
device# show authentication sessions all
-------------------------------------------------------


Port   MAC            IP(v4/v6)            User    VLAN  Auth    Auth    ACL    Session   Age  PAE
       Addr           Addr                 Name          Method  State          Time           State
-------------------------------------------------------------------------------------------------------
-
2/1/25 00aa.aaaa.0000 198.1.1.2            MVDI_1 130   MAUTH   permit  Yes    210       Ena  N/A
2/1/25 00aa.aaaa.0001 fe80::2aa:aaff:feaa  DVDI_1 130   8021.X  permit  Yes    210       Ena  AUTHENTICATED
                      3000::2
                      3000::2
1/1/15 00bb.bbbb.0001 N/A                  DVDI_2 230   8021.X  permit  None   500       Ena  AUTHENTICATED
1/1/10 0010.9400.1101 N/A                  MVDI_2 330   MAUTH   permit  None   410       Ena  N/A
```

The following example displays sessions for a specific interface.

```
device(config)# show authentication sessions ethernet 2/1/25
-------------------------------------------------------------------------
Port   MAC            IP(v4/v6)            User    VLAN  Auth    Auth    ACL    Session   Age  PAE
       Addr           Addr                 Name          Method  State          Time           State
-------------------------------------------------------------------------------------------------------
-
2/1/25 00aa.aaaa.0000 198.1.1.2            MVDI_1  130   MAUTH   permit  Yes    210       Ena  N/A
2/1/25 00aa.aaaa.0001 fe80::2aa:aaff:feaa  DVDI_1  130   8021.X  permit  Yes    210       Ena  AUTHENTICATED
```

The following example displays sessions for a specific stack unit.

```
device(config)# show authentication sessions unit 1
-------------------------------------------------------------
Port   MAC            IP(v4/v6)            User    VLAN  Auth    Auth    ACL    Session   Age  PAE
       Addr           Addr                 Name          Method  State          Time           State
-------------------------------------------------------------------------------------------------------
-
1/1/15 00bb.bbbb.0001 N/A                  DVDI_2  230   8021.X  permit  None   500       Ena  AUTHENTICATED
1/1/10 0010.9400.1101 N/A                  MVDI_2  330   MAUTH   permit  None   410       Ena  N/A
```

The following example displays a brief description of authentication sessions.

```
device# show authentication sessions brief
-----------------------------------------------------------------------------------------------
Port   Number of          Number of          Number of    Untagged            Dynamic
       Attempted Users    Authorized Users   Denied Users VLAN Type           Port ACL MAC DOT1X
       MAC    DOT1X       MAC    DOT1X
-----------------------------------------------------------------------------------------------
1/1/7   0      1           0      1           0      0    RADIUS-VLAN         No
1/1/8   1      0           1      0           0      0    Auth-Default-VLAN   No
1/1/9   0      0           0      0           0      0    Multiple            No
1/1/10  0      0           0      0           0      0    Auth-Default-VLAN   No
```

The following example displays a detailed description of an authentication session.

```
device# show authentication sessions detail ethernet 17/1/1
Auth Session Info (Port 17/1/1, MAC a036.9f6e.1fd2) :
State               : Permitted
Auth Method         : 802.1X                 Auth Mode           : Single Untagged
VLAN Type           : Radius-VLAN            VLAN                : 200
Voice VLAN          : 0                      PVID                : 0
Tagged VLANs        :
User Name           : joe.user@arris.com
Session Time        : 1381                   Reauth Time         : 2220
Idle Timeout        : 120                    Session Timeout     : 0
Acct session ID     : 2                      PCE Index           : 65535
PAE State           : AUTHENTICATED          Age                 : Disabled
```

```
Qos Priority        : 0                     Failure Reason       :
Auth Filter Applied : No                    Tagged               : No
VLAN Add Req State  : Complete              VLAN Del Req State   : Init
Filter Add Req State: Complete              Filter Del Req State : Init
Stale               : No                    Delete Pending       : No
802.1X Enabled      : No                    Session Control      : Self
V4 ACL Applied      : No                    V6 ACL Applied       : No
V4 IN ACL (Session) : acl1                  V4 OUT ACL (Session) : -
V6 IN ACL (Session) : -                     V6 OUT ACL (Session) : -
Client Voice Phone  : No                    Client Wireless AP   : No
802.1X Capable      : Yes
IP Addresses        : 10.176.167.145
V4-IN ACL (Dynamic) : 3928                  V4-OUT ACL (Dynamic) : 0
V6-IN ACL (Dynamic) : 0                     V6-OUT ACL (Dynamic) : 0
V4-IN ACL RefCnt    : 1                     V4-OUT ACL RefCnt    : 0
V6-IN ACL RefCnt    : 0                     V6-OUT ACL RefCnt    : 0
V4 ACL Trap Rule    : Yes                   V6 ACL Trap Rule     : No
Addr Change Count   : 0                     MBV Usage Count      : 1
Radius VLAN RefCnt  : 1

Auth Order          : dot1x, mac-auth       Auth Fail Action     : Restricted VLAN (3)
Auth Timeout Action : Failure               Aging                : Enabled
SG Protection       : Disabled              DOS Protection       : Disabled (limit = 512)
Reauthentication    : Enabled               Reauth Period        : 3600
Reauth Timeout      : 300                   Max Ssessions        : 2
Port Control        : Auto                  Quiet Period         : 60
Supplicant Time     : 30                    Tx Period            : 3
Max Reauth Requests : 2                     Max Frame Retries    : 2
```

Displaying information about user ACLs

You can display information about the currently active user-defined and dynamically applied ACLs on all the interfaces in the stack as shown in the following example.

```
device# show authentication acls all
-------------------------------------------------------------------------------------
Port     MAC Address       V4 Ingress     V4 Egress      V6 Ingress     V6 Egress
-------------------------------------------------------------------------------------
1/1/7    0180.c200.0003    -              -              -              -
1/1/8    0100.c200.0003    10             11             v6in           v6out
1/1/9    0200.c200.0003    100            101            v6in           v6out
```

You can display information about the currently active user-defined and dynamically applied ACLs on a stack unit as shown in the following example.

```
device# show authentication acls unit 1
-------------------------------------------------------------------------------------
Port     MAC Address       V4 Ingress     V4 Egress      V6 Ingress     V6 Egress
-------------------------------------------------------------------------------------
1/1/7    0180.c200.0003    -              -              -              -
1/1/8    0100.c200.0003    10             11             v6in           v6out
1/1/9    0200.c200.0003    100            101            v6in           v6out
```

You can display information about the currently active user-defined and dynamically applied ACLs for a specified interface as shown in the following example.

```
device# show authentication acls ethernet 1/1/9
-------------------------------------------------------------------------------------
Port      MAC Address       V4 Ingress     V4 Egress      V6 Ingress     V6 Egress
-------------------------------------------------------------------------------------
1/1/9     0200.c200.0003    100            101            v6in           v6out
```

Displaying dynamically assigned VLAN information

The **show vlan ethernet** command displays dynamically assigned VLAN information as shown in the following example.

```
device# show vlan ethernet 1/1/7
Total PORT-VLAN entries: 8
Maximum PORT-VLAN entries: 1024
Legend: [Stk=Stack-Id, S=Slot]
PORT-VLAN 10, Name Auth-Default, Priority level0, Spanning tree Off
Untagged Ports: None
Tagged Ports: (U1/M1) 48
Mac-Vlan Ports: (U1/M1) 7 8 9
Monitoring: Disabled
device#sh vlan brief ethernet 1/1/7
Port 1/1/7 is a member of 1 VLANs
VLANs 10
MAC VLANs : 10
Tagged VLANs :
```

Clearing authentication details

You can clear 802.1X and MAC-authentication sessions or statistics counters on individual interfaces or on a range of interfaces as shown in the following examples.

To clear the statistics counters on an interface, enter a command such as the following.

```
device# clear authentication statistics ethernet 1/1/1
```

To clear the statistics counters on a range of interfaces, enter a command such as the following.

```
device# clear authentication statistics ethernet 1/1/1 to 1/1/10
```

To clear the statistics counters on all the interfaces of a stack unit, enter a command such as the following.

```
device# clear authentication statistics unit 3
```

To clear the statistics counters on all the interfaces in the stack, enter a command such as the following.

```
device# clear authentication statistics
```

To clear the sessions for a specific MAC address, enter a command such as the following.

```
device# clear authentication sessions 0000.0034.abd4
```

To clear the sessions on an interface, enter a command such as the following.

```
device# clear authentication sessions ethernet 1/1/1
```

To clear the sessions on a range of interfaces, enter a command such as the following.

```
device# clear authentication sessions ethernet 1/1/1 to 1/1/8
```

To clear the sessions on all the interfaces of stack unit, enter a command such as the following.

```
device# clear authentication sessions unit 1
```

To clear the sessions on all the interfaces of stack, enter a command such as the following.

```
device# clear authentication sessions
```

IPsec

IPsec overview

Internet Protocol security (IPsec) is a suite of protocols that provide secure communication between devices at the network layer (Layer 3) across public and private networks.

NOTE
IPsec is only supported on the Ruckus ICX 7450 switch. To enable IPsec functionality, an ICX7400-SERVICE-MOD module must be installed on the device or stack. For further information about the installation procedure, refer to the *Ruckus ICX 7450 Switch Hardware Installation Guide*.
ICX7400-SERVICE-MOD modules are not supported in Ruckus ICX 7450 devices used as port extender (PE) units in a Campus Fabric network.

The ICX7400-SERVICE-MOD module supports, with pre-shared authentication, the Suite-B-GCM-128 and Suite-B-GCM-256 user interface suites described in RFC 6379 (https://tools.ietf.org/html/rfc6379) and should interoperate with third-party equipment that supports Suite-B-GCM-128 and Suite-B-GCM-256.

IPsec provides end-to-end security for data traffic by using encryption and authentication techniques that ensure data privacy. Encrypted packets are routed in the same way as ordinary IP packets.

IPsec components include:

- Authentication Header (AH)
- Encapsulating Security Payload (ESP)
- Internet Key Exchange (IKE)

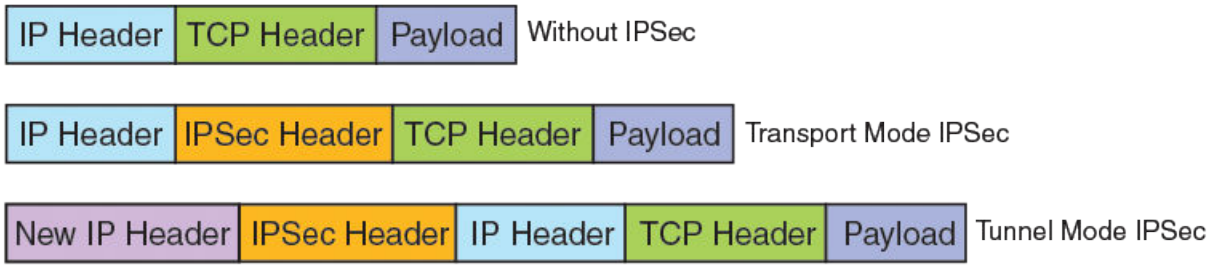
Authentication Header (AH) is a security protocol that provides source authentication and data integrity.

Encapsulating Security Payload (ESP) is a security protocol that provides data confidentiality in addition to the source authentication and data integrity that is also provided by AH. The Ruckus implementation of IPsec uses ESP.

ESP supports two modes of use: transport mode and tunnel mode. In transport mode, an IPsec header is inserted into the IP packet, and the packet payload is encrypted. In tunnel mode, the original IP packet is encrypted as an inner IP payload, and an IPsec header and outer IP header are added so that the IPsec header and encrypted IP packet become the data component of a new and larger IP packet as shown in the following figure.

The ICX 7450 supports ESP in tunnel mode.

FIGURE 13 IPsec tunnel mode versus transport mode



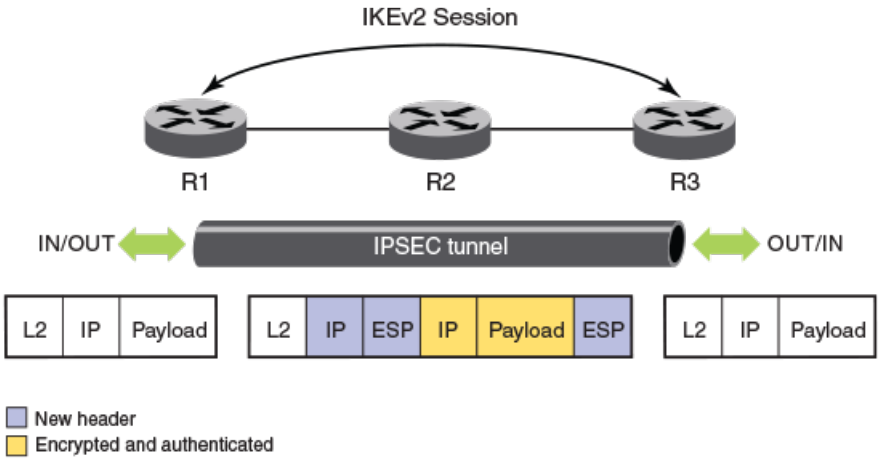
Internet Key Exchange (IKE) is used to establish an IPsec tunnel. IKE performs mutual authentication of peer devices and establishes and maintains a secure channel for communication between the devices.

NOTE
The Ruckus ICX 7450 supports Internet Key Exchange version 2 (IKEv2) only.

A security association (SA) is another important concept in IPsec. The SA is a logically secure relationship between peer devices. Both IKEv2 and IPsec SAs are used to establish an IPsec tunnel.

The Ruckus ICX 7450 supports IPv4 and IPv6 IPsec tunnels.

FIGURE 14 Basic IPsec functionality



The preceding figure shows the secure transfer of IP data between two routers, R1 and R3, over an insecure or public network by using IPsec. First, the tunnel parameters, transform set, and crypto algorithms to be used for encryption and authentication along with associated policy filters are configured on both R1 and R3. IKE negotiations are used to establish the tunnel.

Once the tunnel is up, all packets going out over the tunnel are encrypted, and packets received on the tunnel interface are decrypted.

Acronyms

The following acronyms may be used in this document in the description of IP security.

AES-128-GCM	Advanced Encryption Standard with 128-bit key size in Galois Counter Mode
AES-256-GCM	Advanced Encryption Standard with 256-bit key size in Galois Counter Mode
AES-CBC-128	Advanced Encryption Standard with 128-bit key size in Cipher Block Chaining Mode
AES-CBC-256	Advanced Encryption Standard with 256-bit key size in Cipher Block Chaining Mode
AH	Authentication Header protocol
DH	Diffie-Hellman
DPD	Dead Peer Detection
ESP	Encapsulating Security Payload protocol
IKEv2	Internet Key Exchange protocol, version 2
IPsec	IP security
SA	Security Association
SAD	Security Association Database
SPD	Security Policy Database
SPI	Security Parameters Index (used to identify a SA)
TOS	Type of Service field in IPv4 header
VTI	Virtual Tunnel Interface

Establishment of an IPsec tunnel

IKEv2 negotiations are used to establish an IPsec tunnel.

There are two phases in the IKEv2 negotiation process.

In Phase 1, the tunnel endpoints exchange proposals for mutual authentication and for securing the communication channel. The IKEv2 protocol is used to dynamically negotiate and authenticate keying material and other security parameters that are required to establish secure communications. A secret shared key for encrypting and decrypting the IKEv2 packets themselves is derived in this phase. When the Phase 1 negotiations are successful, an IKEv2 SA, which contains the negotiated security encryption and keying material, is established. The IKEv2 SA is a secure "control channel" where keys and other information for protecting Phase 2 IKEv2 negotiations are maintained. The IKEv2 SA established in Phase 1 is bidirectional.

In Phase 2, the tunnel endpoints exchange proposals for securing the data that is to be sent over the tunnel. Phase 2 communication is secure. When Phase 2 negotiations are successful, a pair of IPsec SAs is established. An IPsec SA is unidirectional; one IPsec SA is needed for inbound traffic and one for outbound traffic. SAs that are negotiated by using the IKE SA (such as IPsec SAs) are also known as child SAs. The negotiated data path keys are then programmed into the specialized hardware crypto engine, and the tunnel state is set to up. At this point, the user data can be exchanged through the encrypted tunnel.

The set of IPsec parameters that describe an IPsec tunnel connection is known as an IPsec security policy. The IPsec security policy describes how both endpoints use the security services, such as encryption and hash algorithms for secure communication.

Multiple security policies may be defined between IPsec security peers. For example, you can define an IPsec security policy that creates a tunnel between two hosts, and a different IPsec security policy that creates a tunnel between a host and a subnet, or between two subnets. Because multiple tunnels can exist between two peers, multiple IPsec SAs can be active at any time between two peers.

Configuration of an IPsec tunnel

Configuration of an IPsec tunnel includes the configuration of virtual tunnel interfaces (VTIs) at the tunnel endpoints and the configuration of both the IKEv2 and IPsec parameters that are used to establish the tunnel and secure the tunnel traffic.

To configure an IPsec tunnel, you must complete the following tasks at the tunnel endpoints:

- Configure a virtual tunnel interface and set the mode of the tunnel to ipsec.
- Configure the following values (when the default values are not acceptable):
  - Global parameters for IKEv2
  - An IKEv2 proposal
  - An IKEv2 policy
  - An IKEv2 authentication proposal
  - An IKEv2 profile
  - An IPsec proposal
  - An IPsec profile
- Bind the IPsec profile to the VTI by using the **tunnel protection ipsec profile** command.

Configuration of traffic to route over an IPsec tunnel

Traffic is routed over an IPsec tunnel by using static route configuration, a dynamic routing protocol, or by using policy-based routing (PBR).

Ruckus ICX 7450 supports dynamic routing of traffic over an IPsec tunnel by using routes learned by way of Routing Information Protocol (RIP) or Open Shortest Path First (OSPF).

Because IPsec tunnels are established by using virtual tunnel interfaces (VTIs), they can plug into the routing protocol infrastructure of a router. IPsec VTIs impact a change in the packet path based on routing metrics or by toggling the link state of the tunnel. VTIs provide termination points for site-to-site IPsec VPN tunnels and allow them to behave like routable interfaces. VTIs simplify the IPsec configuration and enable common routing capabilities to be used because the endpoint is associated with an actual interface.

Using VTIs offers the following advantages:

- IPsec configuration does not require a static mapping of IPsec sessions to a physical interface.
- IPsec VTIs provide protection for remote access.
- IPsec VTIs simplify encapsulation and do not require the use of crypto maps for IPsec.
- Common interface capabilities can be applied to the IPsec tunnel because there is a routable interface at the tunnel endpoint.
- IPsec VTIs support flexibility for the sending and receiving of unicast encrypted traffic on any physical interface.

Supported algorithms

A number of algorithms are supported for IKEv2 negotiations and data path encryption.

TABLE 30 Algorithms supported for IKEv2 negotiations and data path encryption

	Encryption	Integrity	Pseudorandom function
IKEv2 algorithms	AES-CBC-128	SHA-256	SHA-256
	AES-CBC-256	SHA-384	SHA-384
Data path algorithms	AES-GCM-128	none	none
	AES-GCM-256	none	none

NOTE
Ruckus ICX 7450 switches (with the ICX7400-SERVICE-MOD module installed) interoperate with devices from other manufacturers that support the algorithms listed in the preceding table.

Support for PSK for IKEv2 SAs

Text-based pre-shared key (PSK) authentication is supported for IKEv2 SAs.

PSK values are encrypted using simple base64 encryption. The encrypted form of PSKs is used in both saved and run-time configurations. PSKs are displayed in encrypted format in **show** command output.

Unicast IPv4 over IPsec tunnels

Point-to-point (unicast) communications secured by IPsec are supported for IPv4.

The transmission of encrypted IP packets is supported for IPv4 IPsec tunnels. You can also view IPsec tunnel configuration details and debug configurations.

BGP protocol is supported over IPv4 IPSec tunnel. Also, IPv6 traffic is allowed over IPv4 IPSec tunnel.

IPv6 over IPsec Tunnels

IPv6 over IPsec tunnel is supported in this release.

NOTE
The following combinations are support between payload and tunnel type.

- IPv4 payload over IPv4 tunnel
- IPv6 payload over IPv6 tunnel
- IPv6 payload over IPv4 tunnel

To configure the IPv6 over IPsec tunnels:

1. Enter the **tunnel mode ipsec ipv6** command to configure the underneath IPv6 tunnel source, IPv6 tunnel destination and the IKE policy is IPv6. Option of ipv6 match-address will be introduced to ikev2.

```
device# configure terminal
device(config)# interface tunnel1
device(config-tnif-1)# tunnel source 30::1
device(config-tnif-1)# tunnel destination 40::1
device(config-tnif-1)# tunnel mode ipsec ipv6
device(config-tnif-1)# tunnel protection ipsec profile prof-green
device(config-tnif-1)# ipv6 add 90::1/64
device(config-tnif-1)# exit
```

2. Enter the **ikev2 auth-proposal** command to enter the pre-shared key and complete the Ikev2 authentication proposal.

```
device(config)# ikev2 auth-proposal a12
device(config-ike-auth-proposal-a12)# pre-shared-key 2 $MlVzZCFAbg==
device(config-ike-auth-proposal-a12)# exit
```

3. Enter the **ipsec proposal** command to configure the IPsec proposal, for example, to specify an encryption algorithm as shown.

```
device(config)#ipsec proposal a12
device(config-ipsec-proposal-a12)# encryption-algorithm aes-gcm-128
device(config-ipsec-proposal-a12)# exit
```

4. Configure the Ikev2 policy with address match.

```
device(config)# ikev2 policy a12
device(config-ike-policy-a12)# proposal a12
device(config-ike-policy-a12)# match address-local 2001:100::1/64
device(config-ike-policy-a12)#ipsec profile a12
device(config-ipsec-profile-a12)# description a12
device(config-ipsec-profile-a12)# proposal a12
device(config-ipsec-profile-a12)# ike-profile a12
device(config-ipsec-profile-a12)# lifetime 1440
device(config-ipsec-profile-a12)# replay-protection
device(config-ipsec-profile-a12)# exit
```

5. Configure the Ikev2 profile.

```
device(config-ipsec-profile-a12)# ikev2 profile a12
device(config-ipsec-profile-a12)# description ikeprofile12
device(config-ipsec-profile-a12)# authentication a12
device(config-ipsec-profile-a12)# lifetime 2880
device(config-ipsec-profile-a12)# local-identifier key-id IPSEC-GREEN
device(config-ipsec-profile-a12)# remote-identifier key-id IPSEC-BLUE
device(config-ipsec-profile-a12)# match-identity local key-id IPSEC-GREEN
```

```
device(config-ipsec-profile-a12)# match-identity remote key-id IPSEC-BLUE
device(config-ipsec-profile-a12)# exit
```

6. (Optional) Enter the **show ikev2 proposal**, **show ikev2 profile**, and **show ikev2 policy** commands to display the configuration.

```
device(config)# show ikev2 proposal a12
================================================================
Name                : a12
Encryption          : aes256,aes128
Integrity           : sha384,
Prf                 : sha384,
DH Group            : 384_ECP/Group 20,2048_MODP/Group 14,
Ref Count           : 1

device(config)# show ikev2 profile a12
================================================================
IKEv2 Profile       : a12
Auth Profile        : a12
Match Criteria      :
 Inside VRF         : any
  Local:
   keyid IPSEC-X
  Remote:
   keyid IPSEC-XX
Local Identifier    : keyid IPSEC-GREEN
Remote Identifier   : keyid IPSEC-BLUE
Lifetime            : 172800 sec
Keepalive Check     : 300 sec
Ref Count           : 1

device(config)# show ikev2 policy a12
================================================================
Name                : a12
Vrf                 : any
Local Address/Mask  : 2001:100::1/64
Proposal            : a12
```

IPsec scalability limits

Scalability limits may affect the use of IPsec.

The Ruckus ICX 7450 supports encryption and decryption on the ICX7400-SERVICE-MOD module. Encryption and decryption are hardware-based and are not performed by the software (the IKEv2 key exchanges are performed by the software).

One ICX7400-SERVICE-MOD module can be installed per device or per stack. The ICX7400-SERVICE-MOD module supports a maximum of 100 IPsec IPv4 or IPv6 tunnels. If you try to configure more than 100 tunnels, an error message displays. To configure a new IPsec tunnel when the maximum number of tunnels is already configured, you must remove an existing tunnel.

Limits also apply to the elements that are used to configure IPsec tunnels. The element thresholds are set out in the following table.

TABLE 31 Tunnel element thresholds

Tunnel element	IPsec module threshold
IKE session	20
IKE SA	20 (IKEv2 SAs are bidirectional)
IPsec SA	40 (40 IPsec SAs are needed for 100 IKE tunnels or sessions because IPSec SAs are unidirectional; one for ingress and one for egress)

TABLE 31 Tunnel element thresholds (continued)

Tunnel element	IPsec module threshold
IKE proposal	20
IKE policy	20
IKE profile	20
IPsec proposal	20
IPsec profile	20

Supported features and functionality

The features in the following table are supported for IP unicast communications over IPsec.

TABLE 32 Supported features and functionality for IPsec unicast communications

Feature	IPv4
Static point-to-point tunnel setup between two IP endpoints using IKEv2	Yes
Dead Peer Detection (DPD) using IKEv2 Keep Alive	Yes
Configurable options for tunnel elements, such as IKE SA Lifetime, IKEv2 Keep Alive	Yes
VRF forwarding	
Source and destination addresses of the outer header of the tunneled packet can be: • In a different VRF from the VRF for which the packet is received (including the default global VRF) • In the same VRF that receives the packet	Yes
Configurable VRF for tunnel (outer IP header for ESP packet)	Yes
Multi-VRF forwarding to same remote end point (Multiple IPsec tunnels are set up on the same remote endpoint, one for each inner VRF. Also, a separate IKE session is set up for each IPsec tunnel.)	Yes
ECMP	Yes
LAG	Yes
Protocols	
Encapsulation Security Protocol (ESP) in tunnel mode	Yes
IKE (for tunnel setup and key management)	IKEv2 only
Protocols and features supported over IPsec tunnels	
DHCP relay	Yes
OSPFv2	Yes (OSPFv2 only)
Path MTU discovery	Yes
ping	Yes
RIPv1 and RIPv2	Yes (RIPv1 and RIPv2 only)
SSH	Yes
Telnet	Yes
traceroute	Yes
Cryptography	
Suite B cryptography to provide Top Secret, 256-bit security	Yes

TABLE 32 Supported features and functionality for IPsec unicast communications (continued)

Feature	IPv4
AES-CBC-128 and AES-CBC-256 (confidentiality for IKEv2)	Yes
Diffie-Hellman groups (key exchange). The default DH group is 20. Alternate options include groups 14 and 19. Multiple DH groups may be configured; when multiple groups are configured, the highest DH group configured on both the remote and peer devices is selected.	Yes
HMAC-SHA-256-128 and HMAC-SHA-384-192 for integrity	Yes
HMAC-SHA-256 and HMAC-SHA-384 for pseudorandom function (PRF)	Yes
ICX7400-SERVICE-MOD hardware-based encryption and decryption (no encryption or decryption is done by software)	Yes
AES-GCM-128 and AES-GCM-256 For ESP combined-mode authentication, encryption, and decryption of data	Yes
Line rate support (encryption and decryption at 10 Gbps full duplex)	Yes
Line rate support (authentication at 10 Gbps full duplex)	Yes
Interface to interface traffic forwarding (IP packets)	
Jumbo Frame support (IPv4 MTU of 9159 is supported)	Yes
Physical interface to IPsec tunnel interface	Yes
VE interface to IPsec tunnel interface	Yes
GRE tunnel interface to IPSec tunnel interface	Yes
IPsec IPv4 tunnel interface to IPv4 IPsec tunnel interface (tunnel stitching) **NOTE** End-to-end traffic throughput is 5 Gbps full duplex because traffic to the ICX7400-SERVICE-MOD module is doubled at the router where a tunnel re-enters another tunnel.	Yes
IPsec statistics	
Packet counts and byte counts, including: • Transmit and Receive packet counts for each tunnel • Transmit and Receive byte counts for each tunnel	Yes
IKEv2 packet counters, including IKEv2 Keep Alive packets and IKEv2 error packets	Yes
Traps and syslogs	
Up and Down traps and syslogs	Yes
Syslogs for IKEv2 session-establishment errors	Yes

Unsupported features

Some features are not supported for unicast IP communications over IPsec.

TABLE 33 Unsupported features for unicast communications over IPsec

Feature	Description
Tunnel setup	Because of a hardware limitation and the fact that IPsec tunnels can only carry IP inner packets, IPv4 tunnels cannot be set up if the remote endpoint can only be reached by: • GRE tunnel • IGP shortcut • Manual IPv6 tunnels • 6to4 tunnel • IPsec tunnel • ERSPAN management interface (when the destination is reachable over a tunnel)

TABLE 33 Unsupported features for unicast communications over IPsec (continued)

Feature	Description
ESP in transport mode	ESP in transport mode is not currently supported.

Limitations

There are some limitations that impact the use of IPsec for creating secure tunnels.

The following limitations apply:

- Only one active ICX7400-SERVICE-MOD module is supported in a Ruckus ICX 7450 stack.
- Fragmentation is not supported when traffic is routed over an IPSec tunnel; a fragmented IPsec packet received on an IPv4 IPsec tunnel is dropped because IPsec packets are not re-assembled before decryption.
- GRE and IPsec encapsulation are not performed together for the same flow in the same device.
- When multiple IPSec tunnels are configured on the same device, each IPsec tunnel must have a unique tunnel source, destination, and VRF combination.
- IPsec tunnels are not supported when unicast Reverse Path Forwarding (uRPF) is enabled globally.
- Multicast is not supported over IPsec tunnels.

IKEv2 traps

IKEv2 traps are supported for IKEv2 error conditions.

For further information about supported IKEv2 traps, refer to the *Ruckus FastIron Monitoring Configuration Guide*.

IPsec traps

IPsec traps are supported for IPsec error conditions.

For further information about supported IPsec traps, refer to the *Ruckus FastIron Monitoring Configuration Guide*.

The generation of IPsec traps is controlled. That is, new traps are generated, but repeat traps are limited to one trap per minute.

IPSec over NAT

Network Address Translation (NAT) is a method to remap private IP address to public IP address by modifying the address information in the IP packet. This is done by the transit routers when the traffic pass through them.

Using NAT, you can limit the number of public IP addresses owned by a user. In basic-NAT (one-to-one NAT), IP addresses are mapped one-to-one. However, an effective NAT method is Port Address Translation (PAT), where many private addresses map to a single public IP address.

An IPsec ESP packet does not contain port information like TCP and UDP, and, as a result, a NAT (PAT) device is unable to do mapping and drops the packet. This is overcome by the NAT Traversal (IPsec over NAT) feature, which encapsulates the ESP packet inside a UDP header.

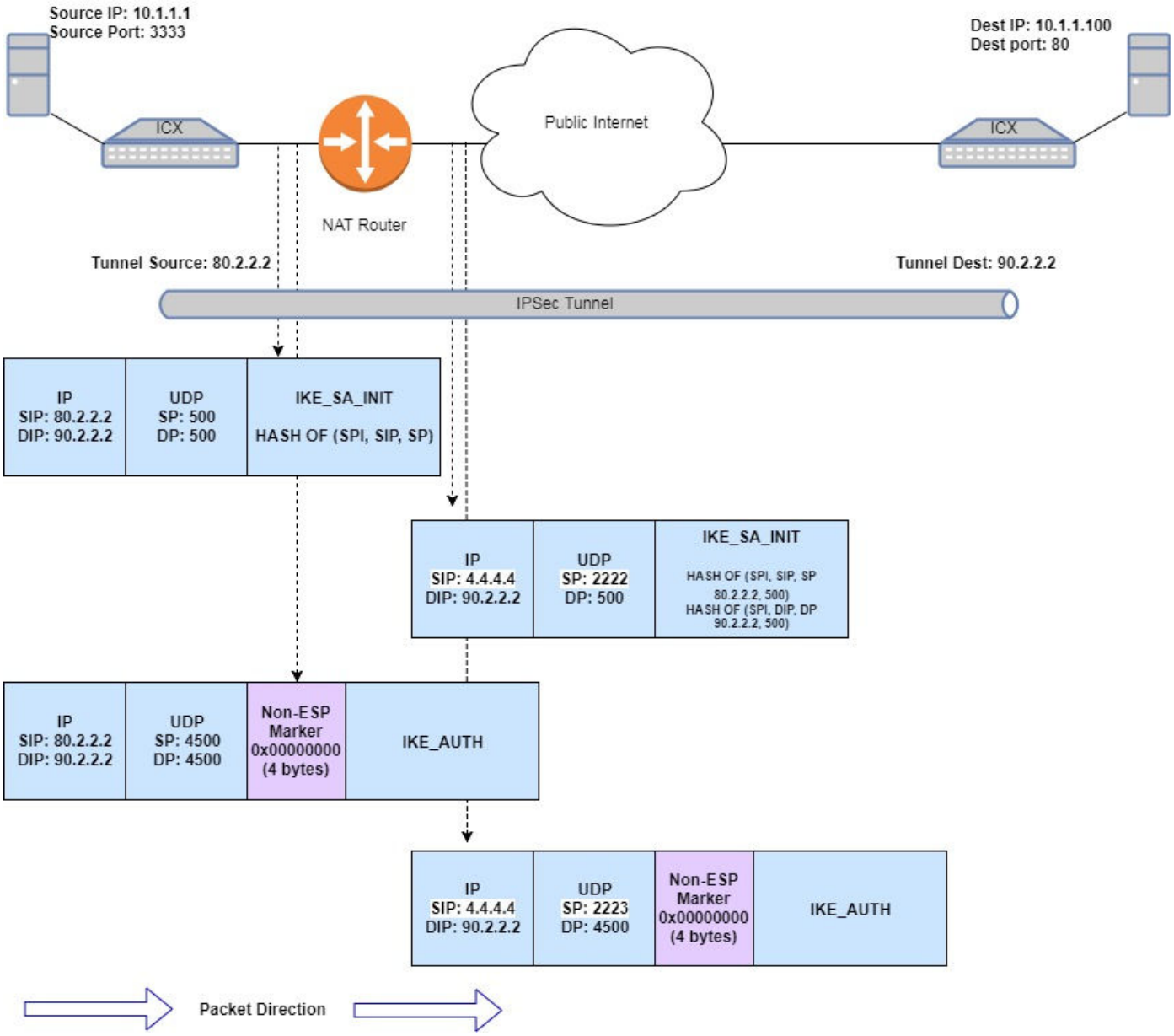
The NAT traversal feature is enabled by default. In FIPS mode, the feature cannot be disabled. In non-FIPS mode, you can disable this feature if necessary.

NOTE
IPsec is supported only on ICX 7450 devices.

How it works

The IKEv2 control protocol discovers any NAT devices between IKE peers by sending new payloads called NAT Discovery (NAT-D) within IKE_SA_INIT messages. Each side sends hashes of SPI, IP address (tunnel addresses), and the port of both IKE peers (source and destination) in the NAT-D payload.

FIGURE 15 IPsec over NAT- Control flow



On the other side, again the hashes of IP address and port are calculated in the packet header. The re-calculated hashes are compared with the hashes received as part of the NAT-D payload. If they are different, it means the packet has undergone NAT. After this negotiation, the IPsec tunnel is established with UDP encapsulation. In UDP, both source and destination ports are set to 4500.

With UDP encapsulation, an ESP packet is treated like a UDP packet, and NAT is applied normally without affecting the ESP packet inside.

Packets are sent periodically to keep the NAT mappings alive. This is configured using the **ikev2 nat keepalive** command. For more information about the command, refer to the *Ruckus FastIron Command Reference*.

NOTE
An SPI value of ZERO should not be used in the SPI field of ESP packets. This value is reserved to distinguish IKE packets from ESP packets.

Configuring IPsec over NAT

The feature is enabled by default. To disable the feature, use the **ikev2 natdisable** command.

NOTE
The **ikev2 nat-disable** command is available only in non-FIPS mode.

```
device(config)# ikev2 nat-disable
```

The NO form of the command enables the feature.

```
device(config)# no ikev2 nat-disable
```

The **show ikev2** commands displays the **NAT-T enabled** status as shown in the following example.

```
device# show ikev2
IKEv2 Global data:
Retry Count          : 5             Max Exchange Time       : 30
Retransmit Interval  : 5             Max SA                  : 256
Max SA In Nego       : 256           Total IPSEC Intf        : 2
Total Peers          : 2             Total IPSEC SA          : 2
Total IKE SA         : 2
NAT-T enabled        : True          NAT-T keepalive time    : 5 sec
```

Downgrade considerations

Software versions that do not support IPsec do not recognize the ICX7400-SERVICE-MOD module interface card.

When system software is downgraded to a version that does not support IPsec, an error message is displayed. To prevent the display of error messages after a downgrade, remove any IPsec-related configuration before the downgrade.

Configuring global parameters for IKEv2

Global Internet Key Exchange version 2 (IKEv2) parameters are configured independently of peer configurations.

When you need to change the default values, use the following commands in global configuration mode to configure global parameters for IKEv2.

For further information and the default values for specific commands, refer to the *Ruckus FastIron Command Reference*.

- Use the **ikev2 exchange-max-time** command to configure the maximum setup time, in seconds, for an IKEv2 message exchange. The following example shows how to set the maximum setup time for IKEv2 message exchange to 50 seconds.

  ```
  device(config)# ikev2 exchange-max-time 50
  ```

- Use the **ikev2 limit max-in-negotiation-sa** command to configure the maximum number of in-negotiation IKEv2 SA sessions. The following example shows how to set the maximum number of in-negotiation IKEv2 SA sessions to 10.

```
device(config)# ikev2 limit max-in-negotiation-sa 10
```

- Use the **ikev2 limit max-sa** command to configure the maximum number of IKEv2 SA sessions. The following example shows how to set the maximum number of IKEv2 SA sessions to 200.

```
device(config)# ikev2 limit max-sa 200
```

- Use the **ikev2 retransmit-interval** command to configure the delay time, in seconds, for resending IKEv2 messages. The following example shows how to set a delay time of 20 seconds.

```
device(config)# ikev2 retransmit-interval 20
```

- Use the **ikev2 retry-count** command to configure the number of attempts to retransmit an IKEv2 message. The following example shows how to set the number of retransmit attempts to 15.

```
device(config)# ikev2 retry-count 15
```

- Use the **show ikev2** command to display the configuration of the global IKEv2 parameters.

```
device# show ikev2

IKEv2 Global data:
Retry Count          : 15           Max Exchange Time       : 50
Retransmit Interval  : 5            Max SA                  : 200
Max SA In Nego       : 10           Total IPSEC Intf        : 0
Total Peers          : 0            Total IPSEC SA          : 0
Total IKE SA         : 0
```

The following example shows how to configure various global IKEv2 parameters.

```
device# configure terminal
device(config)# ikev2 exchange-max-time 50
device(config)# ikev2 limit max-in-negotiation-sa 10
device(config)# ikev2 limit max-sa 200
device(config)# ikev2 retransmit-interval 20
device(config)# ikev2 retry-count 15
```

Configuring an IKEv2 proposal

Internet Key Exchange version 2 (IKEv2) proposal configuration sets parameters that are exchanged in the first phase of IKEv2 peer negotiations. After configuration, an IKEv2 proposal must be attached to an IKEv2 policy for use in IKEv2 negotiations.

There is a default IKEv2 proposal (def-ike-prop) that does not require configuration and has the following settings:

- Encryption algorithm: AES-CBC-256
- PRF algorithm: SHA-384
- Integrity algorithm: SHA-384
- DH group: 20

When the default values are not acceptable, perform the following task to configure an IKEv2 proposal. You only need to complete the steps for settings that you want to change.

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

```
device# configure terminal
```

2. Create an IKEv2 proposal and enter configuration mode for the proposal.

```
device(config)# ikev2 proposal prop_RTB
```

3. Configure an encryption algorithm for the proposal.

```
device(config-ike-proposal-prop_RTB)# encryption aes-cbc-128
```

   This step adds the AES-CBC-128 algorithm to the encryption algorithms configured for prop_RTB. Because the AES-CBC-256 algorithm is configured by default, both the AES-CBC-256 and AES-CBC-128 algorithms are configured for prop_RTB after executing this step. Configuration of multiple encryption algorithms is allowed.

   When you want to configure the AES-CBC-128 algorithm only for the proposal, you must first add the AES-CBC-128 algorithm and then remove the default algorithm by using the **no encryption aes-cbc-256** command.

4. Configure an integrity algorithm for the proposal.

```
device(config-ike-proposal-prop_RTB)# integrity sha256
```

   This step adds the SHA-256 algorithm to the integrity algorithms configured for prop_RTB. Because the SHA-384 algorithm is configured by default, both the SHA-384 and SHA-256 algorithms are configured for prop_RTB after executing this step. Configuration of multiple integrity algorithms is allowed.

   When you want to configure the SHA-256 algorithm only for the proposal, you must first add the SHA-256 algorithm and then remove the default algorithm by using the **no integrity sha384** command.

5. Configure a pseudorandom function (PRF) for the proposal.

```
device(config-ike-proposal-prop_RTB)# prf sha256
```

   This step adds the SHA-256 algorithm to the PRF algorithms configured for prop_RTB. Because the SHA-384 algorithm is configured by default, both the SHA-384 and SHA-256 algorithms are configured for prop_RTB after executing this step. Configuration of multiple PRF algorithms is allowed.

   When you want to configure the SHA-256 algorithm only for the proposal, you must first add the SHA-256 algorithm and then remove the default algorithm by using the **no prf sha384** command.

6. Configure a DH group for the proposal.

```
device(config-ike-proposal-prop_RTB)# dhgroup 19
```

   This step adds DH group 19 to the DH groups configured for prop_RTB. Because DH group 20 is configured by default, both DH groups (19 and 20) are configured for prop_RTB after executing this step. Configuration of multiple DH groups is allowed.

   When you want to configure DH group 19 only for the proposal, you must first add DH group 19 and then remove the default DH group by using the **no dhgroup 20** command.

7. Return to privileged EXEC mode.

```
device(config-ike-proposal-prop_RTB)# end
```

8. Verify the IKEv2 proposal configuration.

```
device# show ikev2 proposal prop_RTB

==========================================================================
Name       : prop_RTB
Encryption : AES-CBC-256,AES-CBC-128
Integrity  : sha384,sha256
PRF        : sha384,sha256
DH Group   : 384_ECP/Group 20,256_ECP/Group 19
Ref Count  : 0
```

The following example shows how to create and configure an IKEv2 proposal named prop-RTB. This example also shows how to remove default configurations; that is, by first configuring an alternate algorithm or DH group and then removing the default configuration.

```
device# configure terminal
device(config)# ikev2 proposal prop_RTB
device(config-ike-proposal-prop_RTB)# encryption aes-cbc-128
device(config-ike-proposal-prop_RTB)# no encryption aes-cbc-256
device(config-ike-proposal-prop_RTB)# integrity sha256
device(config-ike-proposal-prop_RTB)# no intergrity sha384
device(config-ike-proposal-prop_RTB)# prf sha256
device(config-ike-proposal-prop_RTB)# no prf sha384
device(config-ike-proposal-prop_RTB)# dhgroup 19
device(config-ike-proposal-prop_RTB)# no dhgroup 20
device(config-ike-proposal-prop_RTB)# end
```

To use the IKEv2 proposal in IKEv2 negotiations, attach it to an IKEv2 policy by using the **proposal** command in IKEv2 policy configuration mode.

Configuring an IKEv2 policy

Internet Key Exchange version 2 (IKEv2) policy configuration specifies the IKEv2 proposal to be used by an IKEv2 policy and sets match parameters for the policy. An IKEv2 policy is used to protect IKEv2 peer negotiations.

Before configuring an IKEv2 policy, any IKEv2 proposal that is to be associated with the policy must be configured. There is an example at the end of this task that shows all the configuration steps in order.

There is a default IKEv2 policy (def-ike-policy) that does not require configuration. The default policy uses the default IKEv2 proposal (def-ike-prop) and matches:

- All local addresses because a local address to match is not configured for the policy
- Any VRF because a front-door VRF (FVRF) to match is not configured for the policy

  NOTE
  You should not configure overlapping policies. When multiple possible policy matches are configured, the policy that was created most recently is selected.

When the default policy is not acceptable, perform the following task to configure an IKEv2 policy. You only need to complete the steps for settings that you want to change.

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

   ```
   device# configure terminal
   ```

2. Create an IKEv2 policy and enter configuration mode for the policy.

   ```
   device(config)# ikev2 policy pol_RTB
   ```

3. (Optional) An IKEv2 policy must contain at least one IKEv2 proposal. By default, **def-ike-prop** is attached to an IKEv2 policy. Use the **proposal** command to configure an alternate IKEv2 proposal for the IKEv2 policy.

   ```
   device(config-ike-policy-pol_RTB)# proposal prop_RTB
   ```

   This example specifies using an IKEv2 proposal named prop_RTB for the policy.

4. (Optional) When a local match address is not specified, the IKEv2 policy matches all local addresses. Use the **match address-local** command to specify a local IP address as a match parameter for the IKEv2 policy.

```
device(config-ike-policy-pol_RTB)# match address-local 10.3.3.3 255.255.255.0
```

   This example matches the IKEv2 policy pol_RTB based on a specific local IPv4 address (10.3.3.3 255.255.255.0) and the source address of the IPsec tunnel.

5. (Optional) When a front-door VRF (FVRF) to match is not specified, packets that match the local IP addresses specified for the policy are matched to any VRF. Use the **match fvrf** command to specify a front-door VRF for the policy.

```
device(config-ike-policy-pol_RTB)# match fvrf vrf-name example_vrf
```

   This example specifies that the IKEv2 policy pol_RTB is matched when the local IPv4 address and the source address of the IPsec tunnel match and when the base VRF for the IPsec tunnel matches a VRF named example_vrf.

6. Return to privileged EXEC mode.

```
device(config-ike-policy-pol_RTB)# end
```

7. Verify the IKEv2 policy configuration.

```
device# show ikev2 policy pol_RTB

==========================================================================
Name                : pol_RTB
Vrf                 : example_vrf
Local address/Mask  : 10.3.3.3/255.255.255.0
Proposal            : prop_RTB
Ref Count           : 0
```

The following example configures an IKEv2 proposal named prop_RTB. It then configures an IKEv2 policy named pol_RTB using the **proposal** command to attach the IKEv2 proposal (prop_RTB) to the IKEv2 policy.

```
device# configure terminal
device(config)# ikev2 proposal prop_RTB
device(config-ike-proposal-prop_RTB)# encryption aes-cbc-128
device(config-ike-proposal-prop_RTB)# integrity sha256
device(config-ike-proposal-prop_RTB)# prf sha256
device(config-ike-proposal-prop_RTB)# dhgroup 19
device(config-ike-proposal-prop_RTB)# end

device# configure terminal
device(config)# ikev2 policy pol_RTB
device(config-ike-policy-pol_RTB)# proposal prop_RTB
device(config-ike-policy-pol_RTB)# match address-local 10.3.3.3 255.255.255.0
device(config-ike-policy-pol_RTB)# match fvrf vrf-name example_vrf
device(config-ike-policy-pol_RTB)# end
```

Configuring an IKEv2 authentication proposal

Internet Key Exchange version 2 (IKEv2) authentication proposal configuration sets parameters that are used to authenticate IKEv2 peer devices. After configuration, an IKEv2 authentication proposal must be attached to an IKEv2 profile for use in IKEv2 negotiations.

There is a default IKEv2 authentication proposal (def-ike-auth-prop) that does not require configuration and has the following settings:

- Method for local device authentication: pre-shared
- Method for remote device authentication: pre-shared
- Pre-shared key: $QG5HTT1Ebk1TVW5NLWIhVW5ATVMhLS0rc1VA

When the default IKEv2 authentication proposal is not acceptable, perform the following task to configure an IKEv2 authentication proposal.

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

```
device# configure terminal
```

2. Create an IKEv2 authentication proposal and enter configuration mode for the proposal.

```
device(config)# ikev2 auth-proposal auth_blue
```

3. Specify an authentication method for local device authentication.

```
device(config-ike-auth-proposal-auth_blue)# method local pre-shared
```

   This example specifies using a pre-shared key for local device authentication.

4. Specify an authentication method for remote device authentication.

```
device(config-ike-auth-proposal-auth_blue)# method remote pre-shared
```

   This example specifies using a pre-shared key for remote device authentication.

5. (Optional) There is a default pre-shared key that is assigned to an IKEv2 authentication proposal. Use the **pre-shared-key** command to specify an alternate pre-shared key. The following example configures a text-based pre-shared key (ps_key) for the proposal.

```
device(config-ike-auth-proposal-auth_blue)# pre-shared-key ps_key
```

6. Return to privileged EXEC mode.

```
device(config-ike-auth-proposal-auth_blue)# end
```

7. Verify the IKEv2 authentication proposal configuration.

```
device# show ikev2 auth-proposal auth_blue

==========================================================================
Ikev2 Auth-Proposal : auth_blue
Local Auth Method   : pre_shared
Remote Auth Method  : pre_shared
pre-share-key       : $cTJkQ1x4Wnx7UQ==
```

   The encrypted form of the pre-shared key is displayed in the output of the **show ikev2 auth-proposal** command.

The following example creates and configures an IKEv2 authentication proposal named auth_blue.

```
device# configure terminal
device(config)# ikev2 auth-proposal auth_blue
device(config-ike-auth-proposal-auth_blue)# method local pre-shared
device(config-ike-auth-proposal-auth_blue)# method remote pre-shared
device(config-ike-auth-proposal-auth_blue)# pre-shared-key ps_key
device(config-ike-auth-proposal-auth_blue)# end
```

To use the IKEv2 authentication proposal in IKEv2 negotiations, attach it to an IKEv2 profile by using the **authentication** command in IKEv2 profile configuration mode.

Configuring an IKEv2 profile

Internet Key Exchange version 2 (IKEv2) profile configuration sets parameters that are exchanged in the second phase of IKEv2 peer negotiation. An IKEv2 profile specifies match identity criteria and the authentication proposal that is to be applied to an incoming connection. An IKEv2 profile may be used to protect a single VRF or all VRFs.

An IKEv2 session is a unique combination of local IP address, remote IP address, and IKEv2 profile. The IKEv2 profile determines the local identifier that is used for the session.

An IKEv2 profile is applied to an incoming IPsec connection by using match identity criteria presented by incoming IKEv2 connections such as IP address, fully qualified domain name (FQDN), and so on.

For an outgoing connection, the IKEv2 profile is determined by the IPsec profile used for the virtual tunnel interface (VTI).

A complete IKEv2 profile configuration contains a local identity, remote identity, local match identity, and remote match identity. When an IKEv2 profile configuration is incomplete, it is not used.

An IKEv2 VRF matches the forwarding VRF for the VTI.

Before configuring an IKEv2 profile, define and configure the IKEv2 authentication proposal that is to be associated with the profile. There is an example at the end of this task that shows all the configuration steps in order.

There is a default IKEv2 profile (def-ike-profile) that does not require configuration and that has the following settings:

- Authentication proposal: def-ike-auth-prop
- Local identifier address: 0.0.0.0.
- Lifetime period for an IKEv2 security association: 2592000 seconds
- Keepalive interval (the interval between sending IKEv2 messages to detect if a peer is still alive): 300 seconds

  NOTE
  The default IKEv2 profile protects any VRF.

  NOTE
  The method of authentication you select for IKEv2 transactions affects some IKEv2 profile configuration options. When configuring the **local-identifier**, **remote-identifier** and **match-identity**, it is recommended that you select:
  - Distinguished Name (DN), when using PKI-based authentication.
  - Fully Qualified Domain Name (FQDN), when using pre-shared key authentication.

When the default profile is not acceptable, perform the following task to configure an IKEv2 profile.

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

   ```
   device# configure terminal
   ```

2. Create an IKEv2 profile and enter configuration mode for the profile.

   ```
   device(config)# ikev2 profile prof_blue
   ```

3. (Optional) When an IKEv2 profile is created, the default IKEv2 authentication proposal (def-ike-auth-prop) is attached to the profile. Use the **authentication** command to attach an alternate authentication proposal to the profile.

   ```
   device(config-ike-profile-prof_blue)# authentication auth_blue
   ```

   This example attaches the auth-blue authentication proposal to the profile.

4. Specify a local system ID to be sent with the payload during peer negotiation.

```
device(config-ike-profile-prof_blue)# local-identifier address 10.2.2.1
```

5. Specify a remote system ID to be sent with the payload during peer negotiation.

```
device(config-ike-profile-prof_blue)# remote-identifier address 10.3.3.3
```

6. An IKEv2 profile must contain at least one remote identity to match; it is not mandatory to specify a local identity to match. Specify a match identify for the peer device.

   The following example shows how to specify matching on a remote identity (IP address 10.3.3.3).

```
device(config-ike-profile-prof_blue)# match-identity remote address 10.3.3.3
```

   The following example shows how to specify matching on a local identity (IP address 10.3.3.3).

```
device(config-ike-profile-prof_blue)# match-identity local address 10.2.2.1
```

   When multiple match identities are configured, a match occurs when any statement is matched with a peer's local identity or remote identity in remote or local match identity respectively.

7. Specify the name of the VRF that is to be protected.

```
device(config-ike-profile-prof_blue)# protected blue
```

   This example specifies that the IKEv2 profile protects a VRF named blue.

8. Return to privileged EXEC mode.

```
device(config-ike-profile-prof_blue)# end
```

9. Verify the IKEv2 profile configuration.

```
device# show ikev2 profile prof_blue

============================================================================
IKEv2 Profile       : prof_blue
Auth Profile        : auth_blue
Match Criteria      :
 Inside VRF         : blue
  Local:
   address 10.2.2.1
  Remote:
   address 10.3.3.3
Local Identifier    : address 10.2.2.1
Remote Identifier   : address 10.3.3.3
Lifetime            : 2592000 sec
Keepalive Check     : 300 sec
Ref Count           : 0
```

The following example configures an IKEv2 authentication proposal named auth_blue. It then configures an IKEv2 profile named prof_blue using the **authentication** command to attach the IKEv2 authentication proposal to the IKEv2 profile.

```
device# configure terminal
device(config)# ikev2 auth-proposal auth_blue
device(config-ike-auth-proposal-auth_blue)# method local pre-shared
device(config-ike-auth-proposal-auth_blue)# method remote pre-shared
device(config-ike-auth-proposal-auth_blue)# pre-shared-key ps_key
device(config-ike-auth-proposal-auth_blue)# end

device# configure terminal
device(config)# ikev2 profile prof_blue
device(config-ike-profile-prof_blue)# authentication auth_blue
device(config-ike-profile-prof_blue)# local-identifier address 10.2.2.1
device(config-ike-profile-prof_blue)# remote-identifier address 10.3.3.3
device(config-ike-profile-prof_blue)# match-identity local address 10.2.2.1
device(config-ike-profile-prof_blue)# protected blue
device(config-ike-profile-prof_blue)# match-identity remote address 10.3.3.3
device(config-ike-profile-prof_blue)# end
```

To use the IKEv2 profile for outgoing connections, attach it to an IPsec profile by using the **ike-profile** command in IPsec profile configuration mode.

Configuring an IPsec proposal

IPsec proposal configuration sets encryption parameters for IPsec. An IPsec proposal is activated by attaching it to an IPsec profile.

When IPsec is initialized, a default IPsec proposal (def-ipsec-prop) is created with the following settings:

- Transform type: ESP
- Encapsulation mode: Tunnel
- Encryption algorithm: AES-GCM-256 (Both authentication and encryption are performed by this algorithm.)
- Extended sequence numbers: Disabled

When the default proposal is not acceptable, perform the following task to configure an IPsec proposal.

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

   ```
   device# configure terminal
   ```

2. Create an IPsec proposal and enter configuration mode for the proposal.

   ```
   device(config)# ipsec proposal prop_blue
   ```

3. **NOTE**
   (Optional). By default, the encapsulation mode is set to tunnel. Because this is the only mode currently available, this step is not required.

   Specify the encapsulation mode for the proposal.

   ```
   device(config-ipsec-proposal-prop_blue)# encapsulation-mode tunnel
   ```

4. **NOTE**
   (Optional). By default, the transform type is set to ESP. Because this is the only mode currently available, this step is not required.

   Specify the transform type for the proposal.

   ```
   device(config-ipsec-proposal-prop_blue)# transform esp
   ```

5. Specify an encryption algorithm for the proposal.

   ```
   device(config-ipsec-proposal-prop_blue)# encryption-algorithm aes-gcm-128
   ```

   This step adds the AES-GCM-128 algorithm to the encryption algorithms configured for prop_blue. Because the AES-GCM-256 algorithm is configured by default, after executing this step both the AES-GCM-256 and AES-GCM-128 algorithms are configured for prop_blue. Configuration of multiple encryption algorithms is allowed.

   When you want to configure the AES-CBC-128 algorithm only for the proposal, you must first add the AES-CBC-128 algorithm and then remove the default algorithm by using the **no encryption-algorithm aes-cbc-256** command.

   For an IPsec tunnel to come up successfully, IPsec peer devices must be configured with a common encryption algorithm.

6. (Optional) Enable extended sequence number (ESN) as needed.

   NOTE
   ESN is disabled by default. ESN must be enabled for use with replay protection. Replay protection is configured as part of the IPsec profile.
   Replay protection prevents replay attacks by assigning a 64-bit sequence number to each encrypted packet. Processed packets are tracked by their sequence number at the receiving IPsec endpoint and verified against a sliding window of valid sequence numbers.

   NOTE
   Clear IPsec security associations (SAs) for extended sequence numbering to go into effect.

   ```
   device(config-ipsec-proposal-prop_blue)# esn-enable
   ```

7. Return to privileged EXEC mode.

   ```
   device(config-ipsec-proposal-prop_blue)# end
   ```

8. Verify the IPsec proposal configuration.

   ```
   device# show ipsec proposal prop_blue

   ==========================================================================
   Name                : prop_blue
   Protocol            : ESP
   Encryption          : aes-gcm-256,aes-gcm-128
   Authentication      : NULL
   ESN                 : Enable
   Mode                : Tunnel
   Ref Count           : 0
   ```

The following example creates and configures an IPsec proposal named prop_blue.

```
device# configure terminal
device(config)# ipsec proposal prop_blue
device(config-ipsec-proposal-prop_blue)# encapsulation-mode tunnel
device(config-ipsec-proposal-prop_blue)# transform esp
device(config-ipsec-proposal-prop_blue)# encryption-algorithm aes-gcm-128
device(config-ipsec-proposal-prop_blue)# esn-enable
device(config-ipsec-proposal-prop_blue)# end
```

To use the IPsec proposal to encrypt data in an IPsec tunnel, attach it to an IPsec profile by using the **proposal** command in IPsec profile configuration mode.

Configuring an IPsec profile

IPsec profile configuration sets parameters used to encrypt data between IPsec peer devices. After configuration, an IPsec profile is activated by attaching it to a virtual tunnel interface (VTI).

Before configuring an IPsec profile, any IKE profile or IPsec proposal that is to be attached to it must be configured. There is a configuration example at the end of this task that shows all the steps in order.

You can configure an IPsec profile by completing the following task.

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

   ```
   device# configure terminal
   ```

2. Create an IPsec profile and enter configuration mode for the profile.

   ```
   device(config)# ipsec profile prof_blue
   ```

   When an IPsec profile is created, the default IKEv2 profile (def-ike-prof) and the default IPsec proposal (def-ipsec-prop) are automatically attached to the profile.

3. (Optional) Use the **ike-profile** command to attach an alternate IKEv2 profile to the IPsec profile.

   ```
   device(config-ipsec-profile-prof_blue)# ike-profile prof_blue
   ```

   This example attaches the prof-blue IKEv2 profile to the IPsec profile.

4. (Optional) Use the **proposal** command to attach an alternate IPsec proposal to the IPsec profile.

   ```
   device(config-ipsec-profile-prof_blue)# proposal prop_blue
   ```

   This example attaches the prop-blue IPsec proposal to the IPsec profile.

5. (Optional) Configure the lifetime of an IPsec security association (SA).

   The following example shows how to set the IPsec SA lifetime to 240 minutes (14400 seconds).

   ```
   device(config-ipsec-profile-prof_blue)# lifetime 240
   ```

6. (Optional) Configure replay protection.

   NOTE
   Extended sequence numbering (ESN) must also be enabled in the IPsec proposal when replay protection is configured. ESN is disabled by default.
   Replay protection prevents replay attacks by assigning a 64-bit sequence number to each encrypted packet. Processed packets are tracked by their sequence number at the receiving IPsec endpoint and verified against a sliding window of valid sequence numbers.

   NOTE
   Clear IPsec security associations (SAs) for extended sequence numbering to go into effect.

   The following example enables replay protection.

```
device(config-ipsec-profile-prof_blue)# replay-protection
```

7. Return to privileged EXEC mode.

```
device(config-ipsec-profile-prof_blue)# end
```

8. Verify the IPsec profile configuration.

```
device# show ipsec profile prof_blue

==========================================================================
Name                : prof_blue
Ike Profile         : prof_blue
Lifetime            : 14400 sec
Anti-Replay Service : Enabled
DH Group            : None
Proposal            : prop_blue
```

The following example shows how to configure an IKEv2 authorization proposal, an IKEv2 profile, an IPsec proposal and an IPsec profile. The IKEv2 authorization proposal is used in the configuration of the IKEv2 profile. The IKEv2 profile and IPsec proposal are used in the configuration of the IPsec profile.

```
device# configure terminal
device(config)# ikev2 auth-proposal auth_blue
device(config-ike-auth-proposal-auth_blue)# method local pre-shared
device(config-ike-auth-proposal-auth_blue)# method remote pre-shared
device(config-ike-auth-proposal-auth_blue)# pre-shared-key ps_key
device(config-ike-auth-proposal-auth_blue)# end

device# configure terminal
device(config)# ikev2 profile prof_blue
device(config-ike-profile-prof_blue)# authentication auth_blue
device(config-ike-profile-prof_blue)# local-identifier address 10.2.2.1
device(config-ike-profile-prof_blue)# remote-identifier address 10.3.3.3
device(config-ike-profile-prof_blue)# match-identity local address 10.2.2.1
device(config-ike-profile-prof_blue)# protected blue
device(config-ike-profile-prof_blue)# match-identity remote address 10.3.3.3
device(config-ike-profile-prof_blue)# end

device# configure terminal
device(config)# ipsec proposal prop_blue
device(config-ipsec-proposal-prop_blue)# transform esp
device(config-ipsec-proposal-prop_blue)# encryption-algorithm aes-gcm-256
device(config-ipsec-proposal-prop_blue)# esn-enable
device(config-ipsec-proposal-prop_blue)# end

device# configure terminal
device(config)# ipsec profile prof_blue
device(config-ipsec-profile-prof_blue)# ike-profile prof_blue
device(config-ipsec-profile-prof_blue)# proposal prop_blue
device(config-ipsec-profile-prof_blue)# lifetime 240
device(config-ipsec-profile-prof_blue)# replay-protection
device(config-ipsec-profile-prof_blue)# end
```

To activate the IPsec profile, bind it to a virtual tunnel interface (VTI) by using the **tunnel protection ipsec profile** command in tunnel interface configuration mode.

Activating an IPsec profile on a VTI

An IPsec profile is activated by binding it to a virtual tunnel interface (VTI) that is configured as an IPsec VTI.

The IPsec profile must be defined and configured before binding to the VTI. The tunnel interface must be configured also. There is an example at the end of this task that shows the configuration steps in order.

> **NOTE**
> To avoid generating traffic over a stack link, it is recommended that both incoming and outgoing paths to the tunnel endpoints be on the unit where the ICX7400-SERVICE-MOD module is inserted.

You can activate an IPsec profile on a VTI by performing the following task.

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

   ```
   device# configure terminal
   ```

2. Enter configuration mode for an IPsec tunnel.

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

3. Set the mode of the tunnel to IPsec.

```
device(config-tnif-1)# tunnel mode ipsec ipv4
```

4. Specify the IPsec protection profile for the tunnel.

```
device(config-tnif-1)# tunnel protection ipsec profile prof_blue
```

5. Return to privileged EXEC mode.

```
device(config-tnif-1)# end
```

6. Verify that the IPsec profile is attached to the VTI.

```
device# show running-config interface tunnel 1
!
interface tunnel 1
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile prof_blue
!
```

The following example shows how to configure a VTI, set the mode of the tunnel to IPsec, and bind an IPsec profile to the VTI.

```
device# configure terminal
device (config)# interface tunnel 1
device(config-tnif-1)# vrf forwarding blue
device(config-tnif-1)# tunnel source ethernet 1/1/1
device(config-tnif-1)# tunnel destination 10.2.2.1
device(config-tnif-1)# ip address 11.1.1.1/24

device(config-tnif-1)# tunnel mode ipsec ipv4
device(config-tnif-1)# tunnel protection ipsec profile prof_blue
```

Routing traffic over IPsec using static routing

Traffic can be routed over an IPsec tunnel by configuring a static route.

To steer traffic over an IPsec tunnel by configuring a static route, complete the following task.

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

```
device# configure terminal
```

2. Define a static route.

```
device(config)# ip route 10.157.23.0/24 10.4.4.4
```

This example defines a static route with destination IP address 10.157.23.0/24 and where the next-hop address 10.4.4.4 is reachable through the IPsec tunnel.

Alternatively, you can define the static route by specifying the tunnel itself as the outgoing interface.

```
device(config)# ip route 10.157.23.0/24 tunnel 2
```

This example defines a static route with destination IP address 10.157.23.0/24 and specifies tunnel 2 as the outgoing interface.

The following example shows how to configure an IPsec VTI and how to steer traffic over the tunnel by configuring a static route.

```
device# interface tunnel 2
device(config-tnif-2)# vrf forwarding blue
device(config-tnif-2)# tunnel source ethernet 1/1/2
device(config-tnif-2)# tunnel destination 10.2.2.1
device(config-tnif-2)# tunnel mode ipsec ipv4
device(config-tnif-2)# tunnel protection ipsec profile prof-blue
device(config-tnif-2)# ip address 10.4.4.4/24
device(config-tnif-2)# exit

device(config)# ip route vrf blue 10.157.23.0/24 tunnel 2
```

Routing traffic over an IPsec tunnel using PBR

Traffic can be configured to route over an IPsec tunnel by using policy-based routing (PBR) .

Before configuring traffic to route over an IPsec tunnel, the virtual tunnel interface (VTI) must be configured. There is an example at the end of this task that shows the configuration steps in order.

To route traffic over an IPsec tunnel using PBR, complete the following task.

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

   ```
   device# configure terminal
   ```

2. Define an access control list (ACL).

   ```
   device(config)# access-list 99 permit 10.157.23.0 0.0.0.255
   ```

3. Create a route map to accept this traffic and enter route map configuration mode.

   ```
   device(config)# route-map crypto-route permit 99
   ```

   This example creates a route map called "crypto-route".

4. Specify the values to match for the route map.

   ```
   device(config-routemap crypto-route)# match ip address 99
   ```

   This example specifies matching based on the IP address in ACL 99.

5. Configure matching packets to route over the IPsec tunnel.

   ```
   device(config-routemap crypto-route)# set ip next-hop 10.4.4.4
   ```

   This example configures IPsec VTI 10.4.4.4 as the next-hop address for matching packets.

   Alternatively, you can configure the tunnel itself as the next-hop address for matching packets.

   ```
   device(config-routemap crypto-route)# set next-hop-ip-tunnel 2
   ```

   This example configures tunnel 2 as the next-hop address for matching packets.

6. Enter configuration mode on the interface where you want to apply the route map.

   ```
   device(config-routemap crypto-route)# interface ethernet 1/1/3
   ```

   This example enters configuration mode on Ethernet interface 1/1/3.

7. Enable policy-based routing on the interface and specify the route map to be used.

   ```
   device(config-if-e1000-1/1/3)# ip policy route-map crypto-route
   ```

   This example specifies using the "crypto-route" route map for PBR on Ethernet interface 1/1/3.

8. Return to global configuration mode.

```
device(config-if-e1000-1/1/3)# end
```

The following example shows how to configure an IPsec VTI and how to steer traffic over the tunnel by using PBR.

```
device# interface tunnel 1
device(config-tnif-1)# vrf forwarding blue
device(config-tnif-1)# tunnel source ethernet 1/1/1
device(config-tnif-1)# tunnel destination 10.2.2.1
device(config-tnif-1)# tunnel mode ipsec ipv4
device(config-tnif-1)# tunnel protection ipsec profile prof-blue
device(config-tnif-1)# ip address 10.4.4.4/24
device(config-tnif-1)# exit

device(config)# access-list 99 permit 10.157.23.0 0.0.0.255
device(config)# route-map crypto-route permit 99
device(config-routemap crypto-route)# match ip address 99
device(config-routemap crypto-route)# set ip next-hop 10.4.4.1 vrf blue
device(config-routemap crypto-route)# end
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# vrf forwarding blue
device(config-if-e1000-1/1/3)# ip policy route-map crypto-route
device(config-if-e1000-1/1/3)# end
```

Re-establishing SAs

An IKEv2 SA or IPsec SA is re-established after the SA is cleared. When the SA is cleared, any existing child SAs are also cleared and re-established.

- Enter the **clear ikev2 sa** command to clear and re-establish an IKEv2 SA. The following example shows how to clear an IKEv2 SA (and any existing child SAs) by specifying the local IPv4 address for the SA.

```
device# clear ikev2 sa local 10.1.1.1
```

- Enter the **clear ipec sa** command to clear and re-establish an IPsec SA. The following example shows how to clear an IPsec SA by specifying the peer address for the SA.

```
device# clear ipsec sa peer 10.2.2.2
```

Enabling IKEv2 extended logging

IKEv2 syslog messages are enabled by default. You can configure enhanced logging.

By default, IKEv2 syslog messages record when IKEv2 sessions and IPsec sessions go up or down.

Extended logging records additional information, including the following items:

- Reasons packets are discarded
  - Invalid major version
  - Incorrect Security Parameters Index (SPI) value
  - Incorrect Nonce
  - Authentication error
- Reasons for Phase 1 IKEv2 failures
  - Proposal mismatch
  - Authentication error

  - IKE SA overflow
  - Internal error
- Reasons for Phase 2 IKEv2 failures
  - IPsec SA overflow
  - Proposal mismatch
  - Flow mismatch
  - Internal error

Perform these steps to enable extended logging.

1. Enter global configuration mode.

```
device# configure terminal
device(config)#
```

2. To enable logging for IKEv2 packets, enter the following command.

```
device(config)# logging enable ikev2 ikev2-packet
```

3. To enable logging of extended events, enter the following command.

```
device(config)# logging enable ikev2 ikev2-extended
```

The following example enables extended logging of both IKEv2 packets and events.

```
device# configure terminal
device(config)# logging enable ikev2 ikev2-packet
device(config)# logging enable ikev2 ikev2-extended
```

The following example shows an IKEv2 extended event syslog entry for an IKEv2 Phase 1 failure due to a proposal mismatch.

```
SYSLOG: <13> Jul 20 14:36:39 IKEv2: Phase1 failed  Proposal mismatch Source 8:8:8::2 Destination 8:8:8::2
VRF 0 Tunnel 124
```

Disabling traps and syslog messages for IKEv2 and IPsec

You can disable error traps and syslog messages for Internet Key Exchange version 2 (IKEv2) and IPsec. By default, traps and syslog messages for both IKEv2 and IPsec are enabled.

Use the following commands to disable error traps and syslog messages for IKEv2 and IPsec.

- The **no snmp-server enable traps** command disables error traps for IKEv2 and IPsec.

  The following example shows how to disable error traps for IKEv2.

```
device(config)# no snmp-server enable traps ikev2
```

  The following example shows how to disable error traps for IPsec.

```
device(config)# no snmp-server enable traps ipsec
```

- The **no logging enable ikev2** command disables syslog messages for IKEv2.

```
device(config)# no logging enable ikev2
```

NOTE
To disable only extended logging of IKEv2 packets, use the following command:

```
device(config)# no logging enable ikev2 ikev2-packet
```

NOTE
To disable only extended logging of IKEv2 events, use the following command:

```
device(config)# no logging enable ikev2 ikev2-extended
```

- The **no logging enable ipsec** command disables syslog messages for IPsec.

```
device(config)# no logging enable ipsec
```

Displaying IPsec module information

Some show commands can be used to display information about the status of an installed IPsec interface module and about the current utilization of the module.

- Enter the **show module** command to display module status information.

```
device# show module

      Module                                          Status Ports Starting MAC
U1:M1  ICX7450-48 48-port Management Module             OK     48   cc4e.248d.f8d0
U1:M2  ICX7400-4X10GF 4-port 40G Module                OK     4    cc4e.248d.f901
U1:M3  ICX7400-1X40GQ 1-port 40G Module                OK     1    cc4e.248d.f905
U2:M1  ICX7450-48 48-port Management Module             OK     48   cc4e.248e.4990
U2:M2  ICX7400-4X10GF 4-port 40G Module                OK     4    cc4e.248e.49c1
U2:M3  ICX7400-SERVICE-MOD Module                      OK     0
U2:M4  ICX7400-1X40GQ 1-port 40G Module                OK     1    cc4e.248e.49c9
U3:M1  ICX7450-48 48-port Management Module             OK     48   cc4e.248e.4490
U3:M2  ICX7400-4X10GF 4-port 40G Module                OK     4    cc4e.248e.44c1
U3:M3  ICX7400-1X40GQ 1-port 40G Module                OK     1    cc4e.248e.44c5
U3:M4  ICX7400-1X40GQ 1-port 40G Module                OK     1    cc4e.248e.44c9
```

- Enter the **show ipsec card-utilization** command to display information about the current utilization of the IPsec interface module.

```
device# show ipsec card-utilization

IPSEC Module     : 1/4, admin: UP

card-utilization :
 Tx pkt count           : 2181783416   Rx pkt Count            : 2181782549
 Tx pkt/sec             : 30104535     Rx pkt/sec              : 30104483
 Tx byte count          : 208735473958 Rx byte Count           : 306526015230
 Tx bytes/sec           : 4166829344   Rx bytes/sec            : 4890915108
 Encrypt In Utilization : 53.75%       Encrypt Out Utilization : 100.00%
 Decrypt In Utilization : 100.00%      Decrypt Out Utilization : 93.24%
```

Displaying IKEv2 configuration information

Various show commands can be used to display information about IKEv2 configurations.

IKEv2 must be configured before displaying this information.

- Enter the **show ikev2 proposal** command to display information about IKEv2 proposal configurations.

```
device# show ikev2 proposal
================================================================================
Name       : def-ike-prop
Encryption : AES-CBC-256
Integrity  : sha384
PRF        : sha384
DH Group   : 384_ECP/Group 20
Ref Count  : 2
```

- Enter the **show ikev2 policy** command to display information about IKEv2 policy configurations.

```
device# show ikev2 policy
================================================================================
Name                : ike_policy_red
Vrf                 : Default
Local address/Mask  : 0.0.0.0/0.0.0.0
Proposal            : ike_proposal_red
Ref Count           : 0
================================================================================
Name                : def-ike-policy
Vrf                 : Default
Local address/Mask  : 0.0.0.0/0.0.0.0
Proposal            : def-ike-prop
Ref Count           : 0
```

- Enter the **show ikev2 profile** command to display information about IKEv2 profile configurations The following example displays information about a specific IKEv2 profile named ipsec_tunnel_1.

```
device# show ikev2 profile ipsec_tunnel_1

IKEv2 Profile       : ipsec_tunnel_1
Auth Profile        : ipsec_tunnel_1
Match Criteria      :
Inside VRF          : vrf1
Local               : email ipsec_tunnel_1@example.com
Remote              : email ipsec_tunnel_1@example.com
Local Identifier    : email ipsec_tunnel_1@example.com
Remote Identifier   : email ipsec_tunnel_1@example.com
Lifetime            : 2592000 sec
Keepalive Check     : 10 sec
Initial contact     : yes
Ref Count           : 1
```

- Enter the **show ikev2 sa** command to display summary information about IKEv2 security association (SA) configurations.

```
device# show ikev2 sa

Total SA : 4
Active SA: 4    : Constructing SA:0     : Dying SA:0
--------------------------------------------------------------------------------
tnl-id       local            remote        status  vrf(i)       vrf(f)
--------------------------------------------------------------------------------
tnl 18       10.18.3.4        10.18.3.5     active  default-vrf  default-vrf
tnl 22       10.22.3.4        10.22.3.5     active  default-vrf  default-vrf
tnl 19       10.19.3.4        10.19.3.5     active  default-vrf  default-vrf
tnl 20       10.20.3.4        10.20.3.5     active  default-vrf  default-vrf
```

- Enter the **show ikev2 sa detail** command to display detailed information about IKEv2 SA configurations.

```
device# show ikev2 sa detail

Total SA : 1
Active SA: 1   : Constructing SA:0     : Dying SA:0
--------------------------------------------------------------------------------
tnl-id  Local           Remote          Status       Vrf(i)      Vrf(f)
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
tnl 1  10.1.41.1      10.4.41.1      Active       vrf1        vrf2
--------------------------------------------------------------------------------
Role               : Initiator
Local SPI          : 0x6fb19219160c7d71     Remote SPI: 0xde1b24e5764f311e
Profile            : p1
Policy             : ipsec_tunnel_1
Auth Proposal      : p1
```

- Enter the **show ikev2 session** command to display summary information about IKEv2 session configurations.

```
device# show ikev2 session

IKE count:1, Child Sa Count:2
tnl-id       local             remote        status    vrf(i)      vrf(f)
--------------------------------------------------------------------------------
tnl 18       10.18.3.4/500     10.18.3.5/500 active    default-vrf default-vrf
--------------------------------------------------------------------------------
    Encr: aes-cbc-256, Hash: sha384, DH Grp:384_ECP/Group 20, Auth: pre_shared
    PRF: sha384
    Is Initiator: No
    Local spi  : 0xe115847e85ad667b       Remote spi: 0x7bb5ee3b6074a4b4
    Life/Active Time: 2592000/534 sec
    Rekey count Local: 0       Rekey count Remote: 2
Child Sa:
 id 1
       Local selector  0.0.0.0/0 - 255.255.255.255
       Remote selector 0.0.0.0/0 - 255.255.255.255
       ESP SPI IN/OUT: 0xb278/0x7935
       Encryption: aes-gcm-256, ICV Size: 16 octects, Esp_hmac: Null
       Authetication: null  DH Group:none , Mode: tunnel
       Rekey count Local: 0       Rekey count Remote: 2
```

- Enter the **show ikev2 session detail** command to display detailed information about IKEv2 session configurations.

```
device# show ikev2 session detail

IKE count:4, Child Sa Count:8
tnl-id       local             remote        status   vrf(i)      vrf(f)
--------------------------------------------------------------------------------
tnl 18       10.18.3.4         10.18.3.5     active   default-vrf default-vrf
--------------------------------------------------------------------------------
    Encr: aes-cbc-256, Hash: sha384, DH Grp:384_ECP/Group 20, Auth: pre_shared
    PRF: SHA384
    Is Initiator: Yes
    Local spi  : 0xe115847e85ad667b       Remote spi: 0x7bb5ee3b6074a4b4
    Life/Active Time: 2592000/614 sec
    Status Description: active
    Initiator id: address 10.28.3.4    Responder id: address 10.18.3.5
    no Exchange in progress
    next request message id=4
    Keepalive timer: 300 seconds, retry 0
        Total keepalive sent: 2
        Total keepalive received: 0
        Total Bytes sent    : 524   Total Bytes Received   : 672
    Time past since last msg: 14
    NAT-T is not detected
    Rekey count Local: 0       Rekey count Remote: 2
Child Sa:
 id 1
       Local selector  0.0.0.0/0 - 255.255.255.255
```

```
Remote selector 0.0.0.0/0 - 255.255.255.255
ESP SPI IN/OUT: 0xb278/0x7935
Encryption: aes-gcm-256, ICV Size: 16 octects, Esp_hmac: Null
Authetication: null  DH Group:none , Mode: tunnel
Rekey count Local: 0       Rekey count Remote: 2
```

Displaying IPsec configuration information

Various show commands can be used to display IPsec configuration information.

IPsec must be configured before displaying this information.

- Enter the **show ipsec proposal** command to display information about IPsec proposal configurations.

```
device# show ipsec proposal
================================================================================
Name                : def-ipsec-prop
Protocol            : ESP
Encryption          : aes-gcm-256
Authentication      : NULL
ESN                 : Enable
Mode                : Tunnel
Ref Count           : 1
```

- Enter the **show ipsec profile** command to display information about IPsec profile configurations.

```
device# show ipsec profile
================================================================================
Name                : 17
Description         : 17
Ike Profile         : 17
Lifetime            : 28800 sec
Anti-Replay Service : Enabled
DH Group            : None
Proposal            : 17
```

- Enter the **show ipsec sa** command to display summary information about IPsec security association (SA) configurations.

```
device# show ipsec sa

IPSEC Security Association Database is empty.
SPDID(vrf:if) Dir Encap SPI        Destination     AuthAlg  EncryptAlg
IPSEC Security Association Database(child SA pair:2)
  1:tnl 15    OUT IPSEC_ 0x0000a748 10.1.1.15      NULL      aes-gcm-256
  1:tnl 15    IN  IPSEC_ 0x00007e14 10.1.1.15      NULL      aes-gcm-256
  0:tnl 4     OUT IPSEC_ 0x0000476d 10.22.10.103   NULL      aes-gcm-256
  0:tnl 4     IN  IPSEC_ 0x0000c989 10.20.10.101   NULL      aes-gcm-256
```

- Enter the **show ipsec sa address** command to display detailed information about a specific IPsec SA by specifying the local address of the SA.

```
device# show ipsec sa address 10.19.3.4 detail

IPSEC Security Association Database(child SA pair:0)
    interface           : tnl 19
    Local address: 10.3.3.4/500, Remote address: 10.19.3.5/500
    Inner VRF           : vrf1
     Local Identity (addr/mask/prot/port): address(0.0.0.0/0/0/0)
     Remote Identity(addr/mask/prot/port): address(0.0.0.0/0/0/0)
    DF-bit              : clear
    Profile-name        : 19
    DH group            : none
    Direction           : outbound, SPI: 0x6018
    Mode                : tunnel,
    Protocol            : IPSEC_ESP , Encryption : aes-gcm-256 , Authentication : Null
```

```
    ICV size            : 16 bytes
    lifetime(sec)       : Expiring in 243 secs
    Anti-replay service : Disable
    ESN                 : Disable
    Status              : ACTIVE
    Worry Metric        :0
```

Displaying and clearing statistics for IKEv2 and IPsec

Various commands can be used to display and clear statistical information for IKEv2 and IPsec.

- Enter the **show ikev2 statistics** command to display IKEv2 statistics.

```
device# show ikev2 statistics

Total IKEv2 SA Count active: 0
Incoming IKEv2 Requests: Accepted: 0 Rejected: 0
Outgoing IKEv2 Requests: 0
    Accepted: 0 Rejected: 0 Rejected due to no cookie: 0
IKEv2  Packet Statistics:
    Total Packets Received     : 0
    Total Packets Transmitted  : 2
    Total Packets Retransmitted: 0
    Total Failed Transmission  : 0
    Total Pending Packets      : 0
    Total Buffer Failed        : 0
    Total Keepalive Received   : 0
    Total Keepalive Transmitted: 0
IKEv2 Error Statistics:
    Unsupported Payload   : 0        Invalid IKE SPI  : 0
    Invalid Version       : 0        Invalid Syntax   : 0
    Negotiation Timeout   : 0        No Policy        : 0
    No Protection Suite   : 0        Policy Error     : 0
    IKE Packet Error      : 1        Discard Policy   : 0
    Proposal Mismatch     : 0        Invalid Selectors: 0
    Internal Error        : 0        SA Overflow      : 0
    IKE SA Overflow       : 0        IPSEC SA Overflow: 0
    Authentication Failed : 0        Others           : 0
    Number of HW-SPI Add write : 0        Number of HW-SPI Delete
write: 0
```

- Enter the **clear ikev2 statistics** command to reset or clear IKEv2 statistics.

```
device# clear ikev2 statistics
```

- Enter the **show statistics tunnel** command to display tunnel statistics.

```
device# show statistics tunnel

IP GRE Tunnels
      Tunnel Status  Packet Received  Packet Sent  KA recv  KA sent

IPSEC Tunnels
      Tunnel Status  Packet Received  Packet Sent  Bytes Received  Bytes Sent
  9   down/down      0                0            0               0
  10  up/up          50               16442474     7300            9372173444
```

- Enter the **clear statistics tunnel** command to reset or clear tunnel statistics. The following example shows how to clear statistics for tunnel 10.

```
device# clear statistics tunnel 10
```

Configuration example for an IPsec tunnel using default settings (site-to-site VPN)

An IPsec tunnel is configured by binding an IPsec profile to the virtual tunnel interface (VTI) at each end of the IPsec tunnel. When the default settings for the IPsec profile are used, minimal configuration is needed to establish the tunnel.

FIGURE 16 Deployment of IPsec using the default settings

In the following example, Router1 and Router2 are the devices at each end of the tunnel. On each device, an IPsec profile (profA) is created and bound to the VTI by using the **tunnel protection ipsec profile** command.

NOTE
By default, an IPsec profile has the following settings:

- IKEv2 profile: def-ike-profile
- IPsec proposal: def-ipsec-prop

Router1

```
Router1(config)# ipsec profile profA
Router1(config-ipsec-profile-profA)# exit

Router1(config)# Interface tunnel 20
Router1(config-tnif-20)# tunnel mode ipsec ipv4
Router1(config-tnif-20)# tunnel protection ipsec profile profA
Router1(config-tnif-20)# tunnel source 10.1.1.1
Router1(config-tnif-20)# tunnel destination 10.1.1.2
Router1(config-tnif-20)# ip address 10.0.0.1 255.255.255.0
Router1(config-tnif-20)# exit
```

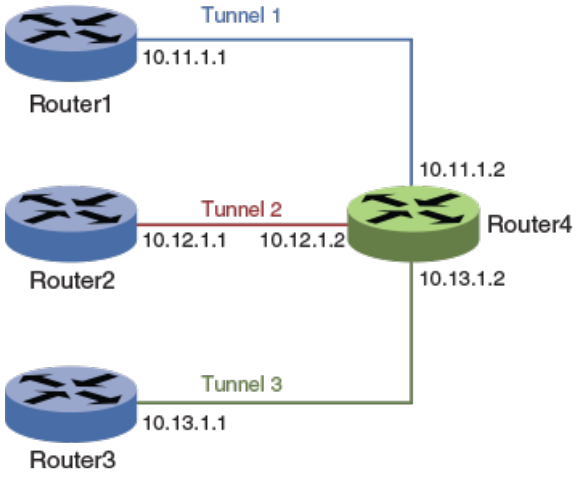
Router2

```
Router2(config)# ipsec profile profA
Router2(config-ipsec-profile-profA)# exit

Router2(config)# Interface tunnel 20
Router2(config-tnif-20)# tunnel mode ipsec ipv4
Router2(config-tnif-20)# tunnel protection ipsec profile profA
Router2(config-tnif-20)# tunnel source 10.1.1.2
Router2(config-tnif-20)# tunnel destination 10.1.1.1
Router2(config-tnif-20)# ip address 10.0.0.2 255.255.255.0
Router2(config-tnif-20)# exit
```

Configuration example for a hub-to-spoke VPN using IPsec

IPsec may be used to secure communications in a hub-to-spoke (tunnel stitching) deployment such as a virtual private network (VPN).

FIGURE 17 Hub-to-spoke deployment of IPsec



NOTE
Tunnel endpoints may be multiple hops away and the base path reachable over any interior gateway protocols such as static routing, RIP, OSPF, BGP and so on.

NOTE
The Ruckus ICX 7450 has a 50 percent performance degradation when used in a tunnel stitching configuration. You should verify if there are platform limitations on any other devices that you use in a tunnel stitching configuration.

In the following configuration example, the IPsec tunnels are running in the user VRF (vrf1) and the base path is in the default VRF.

Router1

```
Router1# configure terminal
Router1(config)# ikev2 proposal ikev2_propA
Router1(config-ike-proposal-ikev2_propA)# exit

Router1(config)# ikev2 auth-proposal ikev2_auth_propA
Router1(config-ike-auth-proposal-ikev2_auth_propA)# pre-shared-key ps_key
Router1(config-ike-auth-proposal-ikev2_auth_propA)# exit

Router1(config)# ikev2 policy ikev2_policyA
Router1(config-ike-policy-ikev2_policyA)# proposal ikev2_proposal
Router1(config-ike-policy-ikev2_policyA)# match address-local 10.100.100.1 255.255.255.255
Router1(config-ike-policy-ikev2_policyA)# exit

Router1(config)# ikev2 profile ikev2_profA
Router1(config-ike-profile-ikev2_profA)# authentication ikev2_auth_propA
Router1(config-ike-profile-ikev2_profA)# local-identifier address 10.1.1.1
Router1(config-ike-profile-ikev2_profA)# remote-identifier address 10.4.4.4
Router1(config-ike-profile-ikev2_profA)# match-identity local address 10.1.1.1
```

```
Router1(config-ike-profile-ikev2_profA)# match-identity remote address 10.4.4.4
Router1(config-ike-profile-ikev2_profA)# exit

Router1(config)# ipsec proposal ipsec_propA
Router1(config-ipsec-proposal-ipsec_propA)# exit

Router1(config)# ipsec profile ipsec_profA
Router1(config-ipsec-profile-ipsec_profA)# proposal ipsec_propA
Router1(config-ipsec-profile-ipsec_profA)# ike-profile ikev2_profA
Router1(config-ipsec-profile-ipsec_profA)# exit

Router1(config)# interface loopback 1
Router1(config-lbif-1)# ip address 10.100.100.1 255.255.255.255
Router1(config-lbif-1)# ip ospf area 0
Router1(config-lbif-1)# exit

Router1(config)# interface tunnel 1
Router1(config-tnif-1)# vrf forwarding vrf1
Router1(config-tnif-1)# tunnel mode ipsec ipv4
Router1(config-tnif-1)# tunnel protection ipsec profile ipsec_profA
Router1(config-tnif-1)# tunnel source loopback 1
Router1(config-tnif-1)# tunnel destination 10.100.100.4
Router1(config-tnif-1)# ip address 10.11.1.1 255.255.255.252
Router1(config-tnif-1)# ip ospf area 0
Router1(config-tnif-1)# exit

Router1(config)# router ospf vrf vrf1
Router1(config-router-ospf-vrf-vrf1)# area 0
Router1(config-router-ospf-vrf-vrf1)# exit

Router1(config)# router ospf
Router1(config-router-ospf-vrf-default-vrf)# area 0
Router1(config-router-ospf-vrf-default-vrf)# end
```

Router2

```
Router2# configure terminal
Router2(config)# ikev2 proposal ikev2_propA
Router2(config-ike-proposal-ikev2_propA)# exit

Router2(config)# ikev2 auth-proposal ikev2_auth_propA
Router2(config-ike-auth-proposal-ikev2_auth_propA)# pre-shared-key ps_key
Router2(config-ike-auth-proposal-ikev2_auth_propA)# exit

Router2(config)# ikev2 policy ikev2_policyA
Router2(config-ike-policy-ikev2_policyA)# proposal ikev2_proposal
Router2(config-ike-policy-ikev2_policyA)# match address-local 10.100.100.2 255.255.255.255
Router2(config-ike-policy-ikev2_policyA)# exit

Router2(config)# ikev2 profile ikev2_profA
Router2(config-ike-profile-ikev2_profA)# authentication ikev2_auth_propA
Router2(config-ike-profile-ikev2_profA)# local-identifier address 10.2.2.2
Router2(config-ike-profile-ikev2_profA)# remote-identifier address 10.4.4.4
Router2(config-ike-profile-ikev2_profA)# match-identity local address 10.2.2.2
Router2(config-ike-profile-ikev2_profA)# match-identity remote address 10.4.4.4
Router2(config-ike-profile-ikev2_profA)# exit

Router2(config)# ipsec proposal ipsec_propA
Router2(config-ipsec-proposal-ipsec_propA)# exit

Router2(config)# ipsec profile ipsec_profA
Router2(config-ipsec-profile-ipsec_profA)# proposal ipsec_propA
Router2(config-ipsec-profile-ipsec_profA)# ike-profile ikev2_profA
Router2(config-ipsec-profile-ipsec_profA)# exit

Router2(config)# interface loopback 1
Router2(config-lbif-1)# ip address 10.100.100.2 255.255.255.255
Router2(config-lbif-1)# ip ospf area 0
Router2(config-lbif-1)# exit
```

```
Router2(config)# interface tunnel 1
Router2(config-tnif-1)# vrf forwarding vrf1
Router2(config-tnif-1)# tunnel mode ipsec ipv4
Router2(config-tnif-1)# tunnel protection ipsec profile ipsec_profA
Router2(config-tnif-1)# tunnel source loopback 1
Router2(config-tnif-1)# tunnel destination 10.100.100.4
Router2(config-tnif-1)# ip address 10.12.1.1 255.255.255.252
Router2(config-tnif-1)# ip ospf area 0
Router2(config-tnif-1)# exit

Router2(config)# router ospf vrf vrf1
Router2(config-router-ospf-vrf-vrf1)# area 0
Router2(config-router-ospf-vrf-vrf1)# exit

Router2(config)# router ospf
Router2(config-router-ospf-vrf-default-vrf)# area 0
Router2(config-router-ospf-vrf-default-vrf)# end
```

Router3

```
Router3# configure terminal
Router3(config)# ikev2 proposal ikev2_propA
Router3(config-ike-proposal-ikev2_propA)# exit

Router3(config)# ikev2 auth-proposal ikev2_auth_propA
Router3(config-ike-auth-proposal-ikev2_auth_propA)# pre-shared-key ps_key
Router3(config-ike-auth-proposal-ikev2_auth_propA)# exit

Router3(config)# ikev2 policy ikev2_policyA
Router3(config-ike-policy-ikev2_policyA)# proposal ikev2_proposal
Router3(config-ike-policy-ikev2_policyA)# match address-local 10.100.100.3 255.255.255.255
Router3(config-ike-policy-ikev2_policyA)# exit

Router3(config)# ikev2 profile ikev2_profA
Router3(config-ike-profile-ikev2_profA)# authentication ikev2_auth_propA
Router3(config-ike-profile-ikev2_profA)# local-identifier address 10.3.3.3
Router3(config-ike-profile-ikev2_profA)# remote-identifier address 10.4.4.4
Router3(config-ike-profile-ikev2_profA)# match-identity local address 10.3.3.3
Router3(config-ike-profile-ikev2_profA)# match-identity remote address 10.4.4.4
Router3(config-ike-profile-ikev2_profA)# exit

Router3(config)# ipsec proposal ipsec_propA
Router3(config-ipsec-proposal-ipsec_propA)# exit

Router3(config)# ipsec profile ipsec_profA
Router3(config-ipsec-profile-ipsec_profA)# proposal ipsec_propA
Router3(config-ipsec-profile-ipsec_profA)# ike-profile ikev2_profA
Router3(config-ipsec-profile-ipsec_profA)# exit

Router3(config)# interface loopback 1
Router3(config-lbif-1)# ip address 10.100.100.3 255.255.255.255
Router3(config-lbif-1)# ip ospf area 0
Router3(config-lbif-1)# exit

Router3(config)# interface tunnel 1
Router3(config-tnif-1)# vrf forwarding vrf1
Router3(config-tnif-1)# tunnel mode ipsec ipv4
Router3(config-tnif-1)# tunnel protection ipsec profile ipsec_profA
Router3(config-tnif-1)# tunnel source loopback 1
Router3(config-tnif-1)# tunnel destination 10.100.100.4
Router3(config-tnif-1)# ip address 10.13.1.1 255.255.255.252
Router3(config-tnif-1)# ip ospf area 0
Router3(config-tnif-1)# exit

Router3(config)# router ospf vrf vrf1
Router3(config-router-ospf-vrf-vrf1)# area 0
Router3(config-router-ospf-vrf-vrf1)# exit
```

```
Router3(config)# router ospf
Router3(config-router-ospf-vrf-default-vrf)# area 0
Router3(config-router-ospf-vrf-default-vrf)# end
```

Router4

Router4 may be any device that supports IPsec. The following example shows how to configure Router4 when the device is a Ruckus ICX 7450 switch.

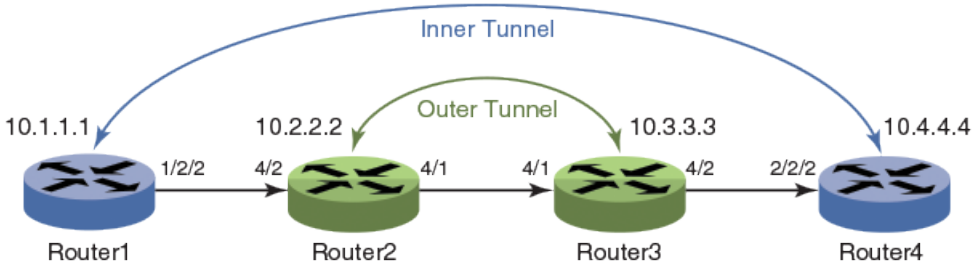
```
Router4# configure terminal
Router4(config)# ikev2 proposal ikev2_propA
Router4(config-ike-proposal-ikev2_propA)# exit

Router4(config)# ikev2 auth-proposal ikev2_auth_propB
Router4(config-ike-auth-proposal-ikev2_auth_propB)# pre-shared-key ps_key
Router4(config-ike-auth-proposal-ikev2_auth_propB)# exit

Router4(config)# ikev2 auth-proposal ikev2_auth_propC
Router4(config-ike-auth-proposal-ikev2_auth_propC)# pre-shared-key ps_key
Router4(config-ike-auth-proposal-ikev2_auth_propC)# exit

Router4(config)# ikev2 auth-proposal ikev2_auth_propD
Router4(config-ike-auth-proposal-ikev2_auth_propD)# pre-shared-key ps_key
Router4(config-ike-auth-proposal-ikev2_auth_propD)# exit

Router4(config)# ikev2 policy ikev2_policyA
Router4(config-ike-policy-ikev2_policyA)# proposal ikev2_propA
Router4(config-ike-policy-ikev2_policyA)# match address-local 10.100.100.4 255.255.255.255
Router4(config-ike-policy-ikev2_policyA)# exit

Router4(config)# ikev2 profile ikev2_profB
Router4(config-ike-profile-ikev2_profB)# authentication ikev2_auth_propB
Router4(config-ike-profile-ikev2_profB)# local-identifier address 10.4.4.4
Router4(config-ike-profile-ikev2_profB)# remote-identifier address 10.1.1.1
Router4(config-ike-profile-ikev2_profB)# match-identity local address 10.4.4.4
Router4(config-ike-profile-ikev2_profB)# match-identity remote address 10.1.1.1
Router4(config-ike-profile-ikev2_profB)# exit

Router4(config)# ikev2 profile ikev2_profC
Router4(config-ike-profile-ikev2_profC)# authentication ikev2_auth_propC
Router4(config-ike-profile-ikev2_profC)# local-identifier address 10.4.4.4
Router4(config-ike-profile-ikev2_profC)# remote-identifier address 10.2.2.2
Router4(config-ike-profile-ikev2_profC)# match-identity local address 10.4.4.4
Router4(config-ike-profile-ikev2_profC)# match-identity remote address 10.2.2.2
Router4(config-ike-profile-ikev2_profC)# exit

Router4(config)# ikev2 profile ikev2_profD
Router4(config-ike-profile-ikev2_profD)# authentication ikev2_auth_propD
Router4(config-ike-profile-ikev2_profD)# local-identifier address 10.4.4.4
Router4(config-ike-profile-ikev2_profD)# remote-identifier address 10.3.3.3
Router4(config-ike-profile-ikev2_profD)# match-identity local address 10.4.4.4
Router4(config-ike-profile-ikev2_profD)# match-identity remote address 10.3.3.3
Router4(config-ike-profile-ikev2_profD)# exit

Router4(config)# ipsec proposal ipsec_propA
Router4(config-ipsec-proposal-ipsec_propA)# exit

Router4(config)# ipsec profile ipsec_profB
Router4(config-ipsec-profile-ipsec_profB)# proposal ipsec_propA
Router4(config-ipsec-profile-ipsec_profB)# ike-profile ikev2_profB
Router4(config-ipsec-profile-ipsec_profB)# exit

Router4(config)# ipsec profile ipsec_profC
Router4(config-ipsec-profile-ipsec_profC)# proposal ipsec_propA
Router4(config-ipsec-profile-ipsec_profC)# ike-profile ikev2_profC
Router4(config-ipsec-profile-ipsec_profC)# exit

Router4(config)# ipsec profile ipsec_profD
Router4(config-ipsec-profile-ipsec_profD)# proposal ipsec_propA
```

```
Router4(config-ipsec-profile-ipsec_profD)# ike-profile ikev2_profD
Router4(config-ipsec-profile-ipsec_profD)# exit

Router4(config)# interface loopback 1
Router4(config-lbif-1)# ip address 10.100.100.4 255.255.255.255
Router4(config-lbif-1)# exit

Router4(config)# interface tunnel 1
Router4(config-tnif-1)# vrf forwarding vrf1
Router4(config-tnif-1)# tunnel mode ipsec ipv4
Router4(config-tnif-1)# tunnel protection ipsec profile ipsec_profB
Router4(config-tnif-1)# tunnel source loopback 1
Router4(config-tnif-1)# tunnel destination 10.100.100.1
Router4(config-tnif-1)# ip address 10.11.1.2 255.255.255.252
Router4(config-tnif-1)# ip ospf area 0
Router4(config-tnif-1)# exit

Router4(config)# interface tunnel 2
Router4(config-tnif-2)# vrf forwarding vrf1
Router4(config-tnif-2)# tunnel mode ipsec ipv4
Router4(config-tnif-2)# tunnel protection ipsec profile ipsec_profC
Router4(config-tnif-2)# tunnel source loopback 1
Router4(config-tnif-2)# tunnel destination 10.100.100.2
Router4(config-tnif-2)# ip address 10.12.1.2 255.255.255.252
Router4(config-tnif-2)# ip ospf area 0
Router4(config-tnif-2)# exit

Router4(config)# interface tunnel 3
Router4(config-tnif-3)# vrf forwarding vrf1
Router4(config-tnif-3)# tunnel mode ipsec ipv4
Router4(config-tnif-3)# tunnel protection ipsec profile ipsec_profD
Router4(config-tnif-3)# tunnel source loopback 1
Router4(config-tnif-3)# tunnel destination 10.100.100.3
Router4(config-tnif-3)# ip address 10.13.1.2 255.255.255.252
Router4(config-tnif-3)# ip ospf area 0
Router4(config-tnif-3)# exit

Router4(config)# router ospf vrf vrf1
Router4(config-router-ospf-vrf-vrf1)# area 0
Router4(config-router-ospf-vrf-vrf1)# exit

Router4(config)# router ospf
Router4(config-router-ospf-vrf-default-vrf)# area 0
Router4(config-router-ospf-vrf-default-vrf)# end
```

Configuration example for an IPsec tunnel in an IPsec tunnel

Double encryption is provided when an IPsec tunnel is configured in another IPsec tunnel.

FIGURE 18 Tunnel in tunnel deployment of IPsec



NOTE
Tunnel endpoints may be multiple hops away and the base path reachable over any interior gateway protocol such as static routing, RIP, OSPF, BGP and so on.

In the following configuration example, the inner tunnel is running in the user VRF (vrf1) and the outer tunnel is running in the default VRF.

Router1

```
Router1# configure terminal
Router1(config)# ikev2 proposal ikev2_propA
Router1(config-ike-proposal-ikev2_propA)# exit

Router1(config)# ikev2 auth-proposal ikev2_auth_propA
Router1(config-ike-auth-proposal-ikev2_auth_propA)# pre-shared-key ps_key
Router1(config-ike-auth-proposal-ikev2_auth_propA)# exit

Router1(config)# ikev2 policy ikev2_policyA
Router1(config-ike-policy-ikev2_policyA)# proposal ikev2_propA
Router1(config-ike-policy-ikev2_policyA)# match address-local 10.100.100.1 255.255.255.255
Router1(config-ike-policy-ikev2_policyA)# exit

Router1(config)# ikev2 profile ikev2_profA
Router1(config-ike-profile-ikev2_profA)# authentication ikev2_auth_propA
Router1(config-ike-profile-ikev2_profA)# local-identifier address 10.1.1.1
Router1(config-ike-profile-ikev2_profA)# remote-identifier address 10.4.4.4
Router1(config-ike-profile-ikev2_profA)# match-identity local address 10.1.1.1
Router1(config-ike-profile-ikev2_profA)# match-identity remote address 10.4.4.4
Router1(config-ike-profile-ikev2_profA)# exit

Router1(config)# ipsec proposal ipsec_propA
Router1(config-ipsec-proposal-ipsec_propA)# exit

Router1(config)# ipsec profile ipsec_profA
Router1(config-ipsec-profile-ipsec_profA)# proposal ipsec_propA
Router1(config-ipsec-profile-ipsec_profA)# ike-profile ikev2_profA
Router1(config-ipsec-profile-ipsec_profA)# exit

Router1(config)# interface loopback 1
Router1(config-lbif-1)# ip address 10.100.100.1 255.255.255.255
Router1(config-lbif-1)# ip ospf area 0
Router1(config-lbif-1)# exit
```

```
Router1(config)# interface tunnel 1
Router1(config-tnif-1)# vrf forwarding vrf1
Router1(config-tnif-1)# tunnel mode ipsec ipv4
Router1(config-tnif-1)# tunnel protection ipsec profile ipsec_profA
Router1(config-tnif-1)# tunnel source loopback 1
Router1(config-tnif-1)# tunnel destination 10.100.100.4
Router1(config-tnif-1)# ip address 10.11.1.1 255.255.255.252
Router1(config-tnif-1)# ip ospf area 0
Router1(config-tnif-1)# exit

Router1(config)# router ospf vrf vrf1
Router1(config-router-ospf-vrf-vrf1)# area 0
Router1(config-router-ospf-vrf-vrf1)# exit

Router1(config)# router ospf
Router1(config-router-ospf-vrf-default-vrf)# area 0
Router1(config-router-ospf-vrf-default-vrf)# end
```

Router2

Router2 may be any device that supports IPsec. The following example shows how to configure Router2 when the device is a Ruckus ICX 7450.

```
Router2# configure terminal
Router2(config)# ikev2 proposal ikev2_propA
Router2(config-ike-proposal-ikev2_propA)# exit

Router2(config)# ikev2 auth-proposal ikev2_auth_propA
Router2(config-ike-auth-proposal-ikev2_auth_propA)# pre-shared-key ps_key
Router2(config-ike-auth-proposal-ikev2_auth_propA)# exit

Router2(config)# ikev2 policy ikev2_policyA
Router2(config-ike-policy-ikev2_policyA)# proposal ikev2_propA
Router2(config-ike-policy-ikev2_policyA)# match address-local 10.100.100.2 255.255.255.255
Router2(config-ike-policy-ikev2_policyA)# exit

Router2(config)# ikev2 profile ikev2_profA
Router2(config-ike-profile-ikev2_profA)# authentication ikev2_auth_propA
Router2(config-ike-profile-ikev2_profA)# local-identifier address 10.2.2.2
Router2(config-ike-profile-ikev2_profA)# remote-identifier address 10.3.3.3
Router2(config-ike-profile-ikev2_profA)# match-identity local address 10.2.2.2
Router2(config-ike-profile-ikev2_profA)# match-identity remote address 10.3.3.3
Router3(config-ike-profile-ikev2_profA)# exit

Router2(config)# ipsec proposal ipsec_propA
Router2(config-ipsec-proposal-ipsec_propA)# exit

Router2(config)# ipsec profile ipsec_profA
Router2(config-ipsec-profile-ipsec_profA)# proposal ipsec_propA
Router2(config-ipsec-profile-ipsec_profA)# ike-profile ikev2_profA
Router2(config-ipsec-profile-ipsec_profA)# exit

Router2(config)# interface loopback 1
Router2(config-lbif-1)# ip address 10.100.100.2 255.255.255.255
Router2(config-lbif-1)# exit

Router2(config)# interface tunnel 1
Router2(config-tnif-1)# tunnel mode ipsec ipv4
Router2(config-tnif-1)# tunnel protection ipsec profile ipsec_profA
Router2(config-tnif-1)# tunnel source loopback 1
Router2(config-tnif-1)# tunnel destination 10.100.100.3
Router2(config-tnif-1)# ip address 10.12.1.1 255.255.255.252
Router2(config-tnif-1)# ip ospf area 0
Router2(config-tnif-1)# exit

Router2(config)# router ospf
Router2(config-router-ospf-vrf-default-vrf)# area 0
Router2(config-router-ospf-vrf-default-vrf)# end
```

Router3

Router3 may be any device that supports IPsec. The following example shows how to configure Router3 when the device is a Ruckus ICX 7450.

```
Router3# configure terminal
Router3(config)# ikev2 proposal ikev2_propA
Router3(config-ike-proposal-ikev2_propA)# exit

Router3(config)# ikev2 auth-proposal ikev2_auth_propA
Router3(config-ike-auth-proposal-ikev2_auth_propA)# pre-shared-key ps_key
Router3(config-ike-auth-proposal-ikev2_auth_propA)# exit

Router3(config)# ikev2 policy ikev2_policyA
Router3(config-ike-policy-ikev2_policyA)# proposal ikev2_propA
Router3(config-ike-policy-ikev2_policyA)# match address-local 10.100.100.3 255.255.255.255
Router3(config-ike-policy-ikev2_policyA)# exit

Router3(config)# ikev2 profile ikev2_profA
Router3(config-ike-profile-ikev2_profA)# authentication ikev2_auth_propA
Router3(config-ike-profile-ikev2_profA)# local-identifier address 10.3.3.3
Router3(config-ike-profile-ikev2_profA)# remote-identifier address 10.2.2.2
Router3(config-ike-profile-ikev2_profA)# match-identity local address 10.3.3.3
Router3(config-ike-profile-ikev2_profA)# match-identity remote address 10.2.2.2
Router3(config-ike-profile-ikev2_profA)# exit

Router3(config)# ipsec proposal ipsec_propA
Router3(config-ipsec-proposal-ipsec_propA)# exit

Router3(config)# ipsec profile ipsec_profA
Router3(config-ipsec-profile-ipsec_profA)# proposal ipsec_propA
Router3(config-ipsec-profile-ipsec_profA)# ike-profile ikev2_profA
Router3(config-ipsec-profile-ipsec_profA)# exit

Router3(config)# interface loopback 1
Router3(config-lbif-1)# ip address 10.100.100.3 255.255.255.255
Router3(config-lbif-1)# exit

Router3(config)# interface tunnel 1
Router3(config-tnif-1)# tunnel mode ipsec ipv4
Router3(config-tnif-1)# tunnel protection ipsec profile ipsec_profA
Router3(config-tnif-1)# tunnel source loopback 1
Router3(config-tnif-1)# tunnel destination 10.100.100.2
Router3(config-tnif-1)# ip address 10.12.1.2 255.255.255.252
Router3(config-tnif-1)# ip ospf area 0
Router3(config-tnif-1)# exit

Router3(config)# router ospf
Router3(config-router-ospf-vrf-default-vrf)# area 0
Router3(config-router-ospf-vrf-default-vrf)# end
```

Router4

```
Router4# configure terminal
Router4(config)# ikev2 proposal ikev2_propA
Router4(config-ike-proposal-ikev2_propA)# exit

Router4(config)# ikev2 auth-proposal ikev2_auth_propA
Router4(config-ike-auth-proposal-ikev2_auth_propA)# pre-shared-key ps_key
Router4(config-ike-auth-proposal-ikev2_auth_propA)# exit

Router4(config)# ikev2 policy ikev2_policyA
Router4(config-ike-policy-ikev2_policyA)# proposal ikev2_propA
Router4(config-ike-policy-ikev2_policyA)# match address-local 10.100.100.4 255.255.255.255
Router4(config-ike-policy-ikev2_policyA)# exit

Router4(config)# ikev2 profile ikev2_profA
Router4(config-ike-profile-ikev2_profA)# authentication ikev2_auth_propA
```

```
Router4(config-ike-profile-ikev2_profA)# local-identifier address 10.4.4.4
Router4(config-ike-profile-ikev2_profA)# remote-identifier address 10.1.1.1
Router4(config-ike-profile-ikev2_profA)# match-identity local address 10.4.4.4
Router4(config-ike-profile-ikev2_profA)# match-identity remote address 10.1.1.1
Router4(config-ike-profile-ikev2_profA)# exit

Router4(config)# ipsec proposal ipsec_propA
Router4(config-ipsec-proposal-ipsec_propA)# exit

Router4(config)# ipsec profile ipsec_profA
Router4(config-ipsec-profile-ipsec_profA)# proposal ipsec_propA
Router4(config-ipsec-profile-ipsec_profA)# ike-profile ikev2_profA
Router4(config-ipsec-profile-ipsec_profA)# exit

Router4(config)# interface loopback 1
Router4(config-lbif-1)# ip address 10.100.100.4 255.255.255.255
Router4(config-lbif-1)# ip ospf area 0
Router4(config-lbif-1)# exit

Router4(config)# interface tunnel 1
Router4(config-tnif-1)# vrf forwarding vrf1
Router4(config-tnif-1)# tunnel mode ipsec ipv4
Router4(config-tnif-1)# tunnel protection ipsec profile ipsec_profA
Router4(config-tnif-1)# tunnel source loopback 1
Router4(config-tnif-1)# tunnel destination 10.100.100.1
Router4(config-tnif-1)# ip address 10.11.1.2 255.255.255.252
Router4(config-tnif-1)# ip ospf area 0
Router4(config-tnif-1)# exit

Router4(config)# router ospf vrf vrf1
Router4(config-router-ospf-vrf-vrf1)# area 0
Router4(config-router-ospf-vrf-vrf1)# exit

Router4(config)# router ospf
Router4(config-router-ospf-vrf-default-vrf)# area 0
Router4(config-router-ospf-vrf-default-vrf)# end
```

PKI support for IPsec

Public Key Infrastructure (PKI) provides certificate management to support secured communication for security protocols such as IP security (IPsec).

A PKI is composed of the following entities:

- Peers communicating on a secure network.
- At least one Certificate Authority (CA) that grants and maintains certificates.
- Digital certificates, which contain information such as the certificate validity period, peer identity information, encryption keys that are used for secure communications, and the signature of the issuing CA.
- An optional registration authority (RA) to offload the CA by processing enrollment requests.
- A distribution mechanism such as Lightweight Directory Access Protocol (LDAP) for certificate revocation lists (CRLs).

PKI provides customers with a scalable, secure mechanism for distributing, managing, and revoking encryption and identity information in a secured data network. Every entity participating in the secured communications is enrolled in the PKI, a process where the device generates a key pair (one private key and one public key) using an asymmetric encryption algorithm and has their identity validated by a trusted entity (also known as a CA or trust point).

When an entity enrolls in PKI, it is granted a digital certificate that is issued by a CA. When peers must negotiate a secured communication session, they exchange digital certificates. Based on the information in the certificate, a peer can validate the identity of another peer and establish an encrypted session with the public keys contained in the certificate.

Certificates

A public key certificate (also known as a digital certificate or identity certificate) is an electronic document used to prove ownership of a public key.

The certificate includes information about the key, information about its owner's identity, and the digital signature of an entity that has verified the certificate's contents are correct. If the signature is valid, and the person examining the certificate trusts the signer, then they know they can use that key to communicate with its owner. Certificates have a finite lifetime, defined by the start and end time within the certificate. The certificate is invalid if it is outside of that lifetime.

- CA certificates are further classified into three classes: cross-certificates, self-issued certificates, and self-signed certificates. Cross-certificates are CA certificates in which the issuer and subject are different entities. Cross-certificates describe a trust relationship between the two CAs. Self-issued certificates are CA certificates in which the issuer and subject are the same entity. Self-issued certificates are generated to support changes in policy or operations. Self-signed certificates are self-issued certificates where the digital signature may be verified by the public key bound into the certificate. Self-signed certificates are used to convey a public key for use to begin certification paths.
- End entity certificates are issued to subjects that are not authorized to issue certificates.

Certificate Authority

Certificate Authority (CA) is an authority in a network that issues and manages security credentials and public keys for message encryption. As part of a Public Key Infrastructure (PKI), a CA checks with a registration authority (RA) to verify information provided by the requestor of a digital certificate. If the RA verifies the requestor's information, the CA can then issue a certificate. The CA also specifies the validity periods of certificates and revokes certificates as needed by publishing CRLs or using Online Certificate Status Protocol (OCSP).

The end entity can choose the revocation type of interest through configuration. OCSP is used for obtaining revocation status of a certificate. When an end entity receives a peer certificate, it sends an OCSP request (over HTTP) to the OCSP server (responder) to know the revocation status of the certificate. The OCSP responder replies back with a signed OCSP Response stating whether the certificate is Good, Revoked or Unknown. If it is not able to process the OCSP request, it reports appropriate errors. The OCSP response can have additional extensions (like OCSPSigning bit) to help customize a particular PKI scheme. If expected extensions are not available in the OCSP response, the end entity can refuse to accept a peer connection.

OCSP Responder running on a Linux device may or may not accept OCSP request received through the HTTP GET method. In such cases, OCSP requests must be sent using the HTTP Post method.

The **revocation-check ocsp** command is used to set OCSP as the revocation type.

```
device(config-pki-trustpoint-trust1)# revocation-check ocsp
```

The **ocsp http post** command is used to configure the HTTP post method.

```
device(config-pki-trustpoint-trust1)# ocsp http post
```

Certificate Revocation List

A Certificate Revocation List (CRL) is a list of certificates signed by the CA that are prematurely invalid.

A periodic CRL timer runs, and each time it expires, it dumps the entire list of revocation information. The revocation check is performed when the CRL information is downloaded for the first time. When the subsequent timer expires, the revocation check is not performed unless the tunnels are forced to re-negotiate.

The **revocation-check crl** command is used to set crl as revocation type.

```
device(config-pki-trustpoint-trust1)# revocation-check crl
```

The **show pki crls** command displays the downloaded revocation information.

```
device# show pki crl < trustpoint_name >
```

The **clear pki crl** command is used to clear the downloaded revocation information.

```
device(config)# clear pki crl < trustpoint_name >
```

The **pki export crl** command is used to export the CRL file of a given trustpoint. The following example exports the CRL for the trustpoint trust 1 to the file crl_file.

```
device(config)# pki export crl trust1 url crl_file
```

CRL Distribution Point

The CRL Distribution Point (CDP) is used to retrieve a CA's latest CRL. The CDP is usually located on an LDAP server or HTTP (web) server and is normally expressed as an ldap://host/dir or http://host/path URL.

Distinguished Name

Distinguished Name (DN) is the set of fields and values that uniquely define a certificate and VPN gateway or RAS VPN client identity. This is sometimes called the "Subject" of the certificate. The DN identity can be used as the IKE ID.

Entity

An entity is an end user of PKI products or services, such as a person, an organization, a device such as a switch or router, or a process running on a computer.

Lightweight Directory Access Protocol

Lightweight Directory Access Protocol (LDAP) is used for accessing and managing PKI information. An LDAP server stores user information and digital certificates from the RA server and provides directory navigation service. From an LDAP server, an entity can retrieve local and CA certificates of its own as well as certificates of other entities.

PKI repository

A PKI repository can be a Lightweight Directory Access Protocol (LDAP) server or a common database. It stores and manages information such as certificate requests, certificates, keys, CRLs, and logs while providing a simple query function.

Registration authority

A registration authority (RA) is an authority in a network that verifies user requests for a digital certificate and tells the Certificate Authority (CA) to issue it. An RA can implement functions including identity authentication, CRL management, key pair generation and key pair backup. The PKI standard recommends that an independent RA be used for registration management to achieve higher security of application systems.

Requester

A requester is the client of SCEP exchanges, such as exchanges involved in requesting and issuing certificates. The requester typically submits SCEP messages for itself, although it can also submit SCEP messages for peers.

Certificate enrollment using SCEP

Ruckus provides functionality for certificate enrollment through the use of the Simple Certificate Enrollment Protocol (SCEP). Certificate enrollment is an essential PKI operation in which a system requester successfully receives a certificate from the Certificate Authority (CA).

SCEP is designed to support the following PKI operations:

- Certificate Authority (CA) public key distribution
- Registration Authority (RA) public key distribution
- Certificate enrollment
- Certificate query
- Certificate Revocation List (CRL) query
- Online Certificate Status Protocol (OCSP) query

Types of enrollment

In FIPS mode, CRL-related commands are blocked. When a user tries to configure a CRL-related command, a message is logged to indicate that the command is not supported in FIPS mode. There are two basic modes: manual enrollment and automatic enrollment.

In manual mode, the user has alternate means of procuring certificates. In such a case, the required certificates are copied to the system flash. These certificates are then imported into PKI and authenticated.

In automatic (auto-enrollment) mode, all messages between the requester and the CA are managed by the network resources. This mode is used to efficiently enroll many system requesters, and the requester systems usually have a common, templated configuration.

Requirements for requesting a certificate

The following requirements must be satisfied before a requester can request a certificate.

- The requester must have at least one appropriate key pair (for example, an EC key pair). This key pair is used to sign the SCEP pkiMessage, which must be completed before a certificate can be issued.
- The following information must be configured locally on the requester (client).
  - The CA IP address, or fully qualified domain name (FQDN).
  - The CA HTTP Computer Gateway Interface (CGI) script path.
  - The identifying information used to authenticate the CA. This information can be obtained from the user or provided (presented) to the end user for manual authorization during the exchange.
  - The one-time challenge password that is sent as part of the Certificate request. This password is used by the CA to validate the local certificate request before signing.

  NOTE
  Multiple independent configurations that contain this information can be maintained by the requester, if needed, to enable interactions with multiple CAs.

Communications between requesters and the CA

Communications between requesters and the CA are made secure using SCEP Secure Message Objects, which specifies how the PKCS #7 cryptographic message syntax is used to encrypt and sign the data. To ensure that the signing operation can be completed, the client uses an appropriate (valid) local certificate.

The following rules apply to the communications between requesters and the CA.

- If the requester already has a certificate issued by the SCEP server and the server supports certificate renewal, the certificate that was already issued should be used (renewed).
- If the requester does not have a certificate issued by the new CA, but does have credentials from an alternate CA, the certificate issued by the alternate CA may be used.

  NOTE
  In this case, policy settings on the new CA determine whether or not the request can be accepted. It can be beneficial to use a certificate from the old domain as credentials when enrolling with a new administrative domain.

- If the requester does not have an appropriate existing certificate, then a locally generated, self-signed certificate must be used. The self-signed certificate must have the same subject name as the name used in the PKCS #10 request.

Configuring PKI

PKI configuration enables you to set up the elements responsible for managing the keys and certificates used to identify and authenticate system requesters and essential PKI elements (such as CAs).

PKI configuration involves a number of tasks, including creating PKI entities and trustpoints, generating and installing certificate requests, and authenticating PKI elements. Creating an enrollment profile is an optional task.

Configuring an entity Distinguished Name

A certificate is the combination of a public key and the identity information of an entity, where the CA identifies a certificate applicant and the identity information using an entity Distinguished Name (DN).

Enter the **pki-entity** command to configure a PKI entity and enter PKI entity configuration sub-mode. In this mode, you can configure other PKI parameters. The following example configures the following parameters for the PKI entity ruckus-entity: common-name, country name, state name, organization unit-name, organization name, email, and location. In addition, parameters such as IP address, fully qualified domain name (FQDN), location, and subject-alternative-name can be configured for the entity.

```
device# configure terminal
device(config)# pki-entity ruckus-entity
device(config-pki-entity-entity1)# common-name "tester1"
device(config-pki-entity-entity1)# country-name "IN"
device(config-pki-entity-entity1)# state-name "KA"
device(config-pki-entity-entity1)# org-unit-name "FI"
device(config-pki-entity-entity1)# org-name "Ruckus"
device(config-pki-entity-entity1)# email-id "user@ruckuswireless.com"
device(config-pki-entity-entity1)# location "BG"
device(config-pki-entity-entity1)# exit
device(config)#
```

Creating a trustpoint

A CA is called a trustpoint because you implicitly trust its authority. The idea is that by trusting a given self-signed certificate, your PKI system will automatically trust any other certificates signed with that trusted certificate. The configuration of multiple trustpoints is supported, and the system supports configuration of up to 10 trustpoints.

Enter the **pki trustpoint** command to configure a PKI trustpoint and enter trustpoint configuration mode.

```
device(config)# pki trustpoint ruckus
device(config-pki-trustpoint-ruckus)#
```

PKI functionality is configured and setup using the following commands.

```
device(config)# pki entity entity1
device(config-pki-entity-entity1)# common-name ruckus
device(config-pki-entity-entity1)# org-unit-name FI
device(config-pki-entity-entity1)# org-name RUCKUS
device(config-pki-entity-entity1)# state-name KA
device(config-pki-entity-entity1)# country-name IN
device(config-pki-entity-entity1)# email-id user@ruckus.com
device(config-pki-entity-entity1)# location BG
```

The **crypto key generate** command offers different methods of generating key-pairs. The following code sample shows the command options available for key generation.

```
device# configure terminal
device(config)# crypto key generate ?
  dsa       generate dsa key pair
  ec        generate elliptical key pair for PKI
  rsa       generate rsa key pair
  <cr>
device(config)# crypto key generate rsa ?
  label     input rsa label
  modulus   generate rsa key size (1024/2048) in non-FIPS mode or 2048 in
            FIPS/CC mode
  <cr>
```

The following example creates enrollment profile profile1 that is added to the configuration for the trustpoint trust1. The trust1 configuration uses an eckeypair with the label ec_2 that was previously generated with the **crypto key generate ec** command.

```
device# configure terminal
device(config)# pki profile-enrollment profile1
device(config-pki-profile-enrollment-profile1)# authentication-url http://FI-PKI02.englab.ruckus.com/
CertSrv/mscep/mscep.dll
device(config-pki-profile-enrollment-profile1)# enrollment-url http://FI-PKI02.englab.ruckus.com/CertSrv/
mscep/mscep.dll

device(config)#pki trustpoint trust1
device(config-pki-trustpoint-trust1)# enrollment retry-count 3
device(config-pki-trustpoint-trust1)# enrollment retry-period 2
device(config-pki-trustpoint-trust1)# enrollment profile profile1
device(config-pki-trustpoint-trust1)# pki-entity entity1
device(config-pki-trustpoint-trust1)# revocation-check crl
device(config-pki-trustpoint-trust1)# crl-query http://FI-PKI02.englab.ruckus.com/CertEnroll/englab-FI-
PKI02-CA.crl
device(config-pki-trustpoint-trust1)# crl-update-time 1
device(config-pki-trustpoint-trust1)# eckeypair key-label ec_2
device(config-pki-trustpoint-trust1)# fingerprint 89:31:79:bd:50:55:ef:84:7f:0c:ae:9a:5c:12:d7:7b:fa:
3b:d1:d8

device(config)#ikev2 auth-proposal a1
device(config-ike-auth-proposal-a1)# method remote ecdsa384
device(config-ike-auth-proposal-a1)# method local ecdsa384
device(config-ike-auth-proposal-a1)# pki-trustpoint trust1 sign
device(config-ike-auth-proposal-a1)# pki-trustpoint trust1 verify
```

The following example configures the IKEv2 authentication proposal a1, which uses trustpoint trust1 to sign and verify certificates.

```
device(config)# ikev2 auth-proposal a1
device(config-ike-auth-proposal-a1)# method remote ecdsa384
device(config-ike-auth-proposal-a1)# method local ecdsa384
device(config-ike-auth-proposal-a1)# pki-trustpoint trust1 sign
device(config-ike-auth-proposal-a1)# pki-trustpoint trust1 verify
```

The following authentication algorithms can also be configured apart from ecdsa384:

- ecdsa256
- rsa
- rsa2048
- pre-shared

If pre-shared is selected, PKI functionality is not used. Instead, the pre-shared key is used to negotiate with peers, in which case, both endpoints must have the same pre-shared key configured.

The **pki authenticate** command is used to authenticate the trustpoint. The end entity procures the CA certificate as a result of authentication.

```
device(config)# pki authenticate <trustpoint-name>
```

The **pki enroll** command is used to enroll the trustpoint with the CA server. The end entity procures its local certificate with a digital signature from the CA server as a result of enrollment.

```
device(config)# pki enroll <trustpoint-name>
```

The certificates for the end entity can be generated using PKI infrastructure or can be pre-generated and copied offline into the flash of the device.

The following commands are used to import stored certificates from the flash of the device.

```
device(config)# pki import trust1 pem url flash: <root-ca-file-name>.pem
device(config)# pki import trust1 pem url flash: <end-entity-file-name>.pem
device(config)# pki import key ec <key-label> pem url flash: <end-entity-key-file-name>.pem
device(config)# pki import key rsa <key-label> pem url flash: <end-entity-key-file-name>.pem
```

The following commands are available to display information from the PKI database.

```
device# show pki
  certificates            Display pki certificates
  counters                Show PKI counters
  crls                          Show PKI Certification Revocation list if Any
  enrollment-profile      Show PKI enrollment profile.
  entity                  Show PKI entity.
  key                     Show router public keys.
  logging-statistics      Display pki logging statistics
  trustpoint              Show  PKI trustpoint information
```

The following **show pki certificates** commands are used to display the certificate information associated with specific trustpoints in the PKI database.

```
device# show pki certificates trustpoint <trustpoint-name>
device# show pki certificates trustpoint <trustpoint-name> detail
device# show pki certificates local trustpoint <trustpoint-name>
device# show pki certificates local trustpoint <trustpoint-name> detail
```

Configuring CA authentication

Your router authenticates the CA by obtaining the CA self-signed certificate (this certificate contains the public key of the CA). Because the CA signs its own certificate, you should manually authenticate the public key of the CA by contacting the CA administrator when you enter the command to authenticate the CA. The certificate obtained from the CA is saved to the router.

Enter the **pki authenticate** command to configure CA authentication.

```
device(config)# pki authenticate ruckus
```

The no form of this command removes both authentication and enrollment of the trustpoint.

Generating a certificate request

Your router requests certificates from the CA (trustpoint) to be added to each key pair of your router. This enrolls the router on the CA trustpoint. You use a single command to request the certificate. The certificates are saved to the router.

Enter the **pki enroll** command to generate a certificate request (you specify the CA by trustpoint name using the name parameter.

```
device(config)# pki enroll ruckus
```

Use the no form of the pki enroll command to remove the certificates from the router.

Extended Key Usage

Extended Key Usage (EKU) is a method of enforcing the public key of a certificate to be used for a pre-determined set of key purposes.

There can be one or more such key purposes defined. This extension is usually defined by the end entity systems in their certificates to support their security design constraints. When EKU is present in a certificate, it implies that the public key can be used in addition to or in place of the basic purposes listed in the key usage extension. The EKU extension is always tagged as critical.

The feature enables PKI clients like TLS or IKE to include EKU extension in their certificates and also process received EKU enabled certificates from peer and take action (accept or reject a connection). The EKU extension has key purposes as follows:

- Server authentication (OID 1.3.6.1.5.5.7.3.1)
- Client Authentication (OID 1.3.6.1.5.5.7.3.2)
- anyExtendedKeyUsage (OID 2.5.29.37.0)

Every fields are uniquely identified by an OID.

System in FIPS mode.

Upon receiving a peer certificate:

- For an IKE client, if peer certificate contains an EKU without "anyExtendedKeyUsage" set, the certificate is rejected. If the peer certificate does not contain the EKU extension, the certificate is accepted.
- For a TLS client, if peer certificate does not have an EKU extension, or contains an EKU without server authentication set, the certificate is rejected.

System in Non-FIPS mode.

Upon receiving a peer certificate:

- For an IKE client, if peer certificate contains an EKU without "anyExtendedKeyUsage" set, the certificate is rejected. If the peer certificate does not contain the EKU extension, the certificate is accepted.
- For a TLS client, if peer certificate does not have an EKU extension, the certificate is accepted. If the EKU extension is available without server authentication set, the certificate is rejected.

Configuration

The EKU extension is not enabled by default in the certificate. The PKI client (like IKE/TLS) has to provision it explicitly using the **extended-key-usage** command.

```
device(config)#pki trustpoint <trust-point name>
device(config-pki-trustpoint-trust1)#extended-key-usage ?
client-auth            Enable client authentication.
server-auth            Enable server authentication.
any                Enable any extended key
```

The NO form of this command disables the EKU extension.

Creating a PKI enrollment profile

You can create a PKI enrollment profile you can use to efficiently enroll requester systems. When you create a profile, you name the profile and specify the values for the parameters used to enroll requester systems. Once the profile is defined, you can use it to enroll requester systems.

To define an enrollment profile, enter the **pki profile-enrollment** command in global configuration mode. Using this command enters pki-profile mode, which is required to configure the enrollment profile parameters.

Use the no form of this command to delete all information defined in the enrollment profile.

> **NOTE**
> You must specify the authentication and enrollment URLs in the correct form. The URL argument must be in the form http://CA_name, where CA_name is the host Domain Name System (DNS) name or IP address of the CA.

To create a PKI enrollment profile, do the following:

1. Enter the **pki profile-enrollment** command.

   ```
   device(config)# pki profile-enrollment
   ```

2. Use the following parameters to specify values for the enrollment profile:
   - **name**: The name of the profile.
   - **authentication-url** *url-string*: The URL of the certification authority (CA) server you want to receive the authentication requests. Make sure you use the correct form of the URL.
   - (Optional) **authentication-command** *url-string*: The HTTP command that is sent to the certification authority (CA) for authentication.

     ```
     authentication command GET /certs/cacert.der
     ```

   - **enrollment-url** *url-string*: The URL of the certification authority (CA) server you want to receive the enrollment requests. Make sure you use the correct form of the URL.
   - (Optional) **password**: The password for the SCEP challenge used to revoke the requester's current certificate and issue another certificate for auto mode. No default value is configured.

Installing identity certificates

Manually installs (by import) certificates from the flash memory of the CA trustpoint to the system requester (router). You specify the trustpoint by name that issues the certificates the system requester imports. One command is used to specify the CA trustpoint and another command is used to export the already-imported certificates to the CA. Exporting certificates ensures that they can be used again if the router is rebooted.

> **NOTE**
> The trustpoint names you specify using the commands to import and export the certificates must match the name of the trustpoint you specified using the crypto **pki trustpoint** command.

Enter the **pki import command** to specify the trustpoint that issues the certificates the system requester imports.

```
device(config)# pki import
```

Enter the **pki export** command to specify the trustpoint that receives the certificates exported by the system requester.

```
device(config)# pki export
```

Clearing the certificate revocation list (CRL) and PKI counters

Enter the **clear pki crl** command followed by the appropriate trustpoint name to delete the certificate revocation list (CRL) database from the CA (trustpoint). The following example deletes the CRL for the trustpoint trust1.

```
device# configure terminal
device(config)# clear pki crl trust1
```

Enter the **clear pki counters** command to clear all PKI counters.

```
device# configure terminal
device(config)# clear pki counters
```

Enabling PKI logging

Follow these steps to configure PKI logging or PKI extended logging.

1. To configure PKI logging of events, enter the following commands.

   ```
   device# configure terminal
   device(config)# logging enable pki
   ```

2. To configure PKI extended logging, including information on both events and PKI packets, enter the following commands.

   ```
   device# configure terminal
   device(config)# logging enable pki-extended
   ```

3. Enter the **show pki logging-statistics** command to display PKI counters and statistics as shown in the following example.

   ```
   device# show pki logging-statistics
   ------------------------PKI logging statistics-------------------------------
   Type of packet:                    |     TX_PACKETS     |     RX_PACKETS
   -----------------------------------|--------------------|------------------
   enrollment packets:                |       0            |       0
   authentication packets:            |       1            |       1
   revocation check packets:          |     116            |       0
   peer certificate download packets: |       0            |       0
   certificate imports through http:  |       0            |       0
   Note:enrollment packets can be 2x of actual enrollments depends on server settings
   ```

Displaying PKI information

Display PKI information including, certificates, CA status, certificate evocation lists, PKI counters, public keys, and current enrollment profiles, and PKI entities.

Enter the **show pki trustpoint** command to display information on configured trustpoints.

```
device# show pki trustpoint
----------------PKI TRUSTPOINT ENTRY-----------------
CA: trustRSA
Key Information:
The key label is icx_rsa_key
Public-Key: (2048 bit)
Modulus:
00:c5:81:6f:98:aa:f8:e4:a8:2d:d9:f3:d7:d0:e7:
5e:be:59:4b:4c:d0:c9:aa:a8:53:82:dd:2f:df:09:
c1:78:c5:38:63:c3:d7:73:47:ed:43:6c:d6:d1:ed:
99:82:e7:51:c6:03:bc:8e:8f:97:e5:1b:b5:71:a1:
46:f4:a8:b2:bb:6e:61:54:e2:42:1e:63:f8:79:78:
6b:bd:d8:63:67:c1:b7:6f:78:cc:9d:16:42:df:81:
d2:98:24:2b:70:60:10:ec:0e:5c:d9:be:7e:e1:a0:
27:b8:e0:65:73:99:de:18:59:05:e7:7e:df:f1:1e:
ac:ab:33:7a:7e:6e:d5:99:85:95:fc:c8:a7:1f:c3:
d2:43:74:2e:c6:15:80:b6:fc:73:4c:23:30:2a:c1:
26:d0:84:4c:58:96:0b:4c:1c:f0:87:cf:d3:28:68:
0a:65:f7:fd:33:cb:92:c7:d5:8d:df:7b:9b:03:92:
d8:75:03:1c:f6:1b:09:b3:6d:3c:2a:7e:6a:02:10:
21:5c:46:87:46:73:57:7c:66:8f:a4:bb:a4:6b:ae:
30:d2:63:a0:44:44:6b:48:e2:ab:8e:fa:d4:d7:f7:
30:87:c1:11:ac:22:9f:e9:10:52:ee:22:70:c6:f7:
6b:5b:eb:7f:f3:b3:01:a9:d6:25:10:97:1b:9d:7e:
50:51
Exponent: 65537 (0x10001)
Configured Fingerprint for authentication:
D8:BC:F5:94:BA:72:9D:F3:34:77:FD:AA:5B:A2:FD:B6:59:A3:00:27
Enrollment Protocol:SCEP
----------------PKI TRUSTPOINT ENTRY-----------------
CA: trust1
Entity Name: entity1
Common Name: tester1
Organization Name: BRCD
Organization Unit Name: FI
State Name: BC
Country Name: CA
Email: user@brocade.com
Location: BG
Configured Fingerprint for authentication:
d2:52:b6:5a:1d:a2:95:3b:f4:e6:05:33:84:05:97:16:75:15:bf:04
Enrollment Protocol:SCEP
Enrollment Profile: profile1
```

Enter the **show pki certificates trustpoint** command to display information on trustpoints and related certificates.

```
device# show pki certificates trustpoint
----------------PKI TRUSTPOINT CERTIFICATE ENTRY-----------------
CA: trustRSA
Certificate:
Data:
Version: 3 (0x2)
Serial Number:
e2:11:82:3f:37:c2:6f:c0
Signature Algorithm: sha256WithRSAEncryption
Issuer: C=IN, ST=KA, L=Bangalore, O=Ruckus Arris, OU=NEBU, CN=ROOT RSA
Validity
Not Before: Feb 23 05:38:11 2018 GMT
Not After : Feb 23 05:38:11 2023 GMT
Subject: C=IN, ST=KA, L=Bangalore, O=Ruckus Arris, OU=NEBU, CN=ROOT RSA
```

Enter the **show pki certificates local** command to display information about local certificates.

```
device# show pki certificates local
----------------PKI LOCAL CERTIFICATE ENTRY-----------------
CA: trustRSA
Certificate:
Data:
Version: 3 (0x2)
Serial Number: 4100 (0x1004)
Signature Algorithm: sha256WithRSAEncryption
Issuer: C=IN, ST=KA, L=Bangalore, O=Ruckus Arris, OU=NEBU, CN=ROOT RSA
Validity
Not Before: Feb 23 16:19:43 2018 GMT
Not After : Feb 21 16:19:43 2028 GMT
Subject: CN=ICX RSA, ST=KA, C=IN, O=NEBU, OU=Ruckus Arris
```

Enter the **show pki crls** command followed by a trustpoint name to display the current certificate revocation lists on the specified trustpoint (trust1 in the following example).

```
device(config)# show pki crls trust1
```

Enter the **show pki counters** command to display the current PKI counters.

```
device# show pki counters
------------PKI-COUNTERS----------
PKI Sessions Started: 3701
PKI Sessions Ended: 3701
PKI Sessions Active: 0
Successful Validations: 35
Failed Validations: 3855
Bypassed Validations: 0
Pending Validations: 5
CRLs checked: 0
CRL - fetch attempts: 0
CRL - failed attempts: 0
```

Enter the **show pki key mypubkey all** command to display information about the current public keys on the router. You can choose to display only generated keys, manually imported keys, or all keys.

```
device# show pki key mypubkey all
----------------PKI PUBLIC KEY ENTRY-----------------
Public key of generated EC key pair:
The key label is marcia_ec
Public-Key: (384 bit)
pub:
04:61:f6:d4:bf:e0:85:8f:2f:70:e3:79:36:d9:22:
98:ca:3e:6e:10:a3:cd:b9:0a:e9:2d:26:ce:a3:fc:
96:c5:04:f7:28:6b:fa:fb:e1:36:51:4b:05:05:95:
da:e7:14:5f:59:68:16:2b:fc:c7:a0:d6:a0:72:85:
28:dd:54:10:1e:42:51:0d:8e:d7:6b:2f:92:cc:e2:
ac:f6:f5:89:64:da:54:af:b5:26:e1:f6:a5:25:f2:
a9:93:3c:9a:b8:93:5b
ASN1 OID: secp384r1
```

Enter the **show pki enrollment-profile** command followed by the profile name to display information about the specified enrollment profile.

```
device# show pki enrollment-profile profile1
----------------PKI ENROLLMENT PROFILE ENTRY-----------------
Enrollment Profile: profile1
Authentication Command: WINN6C3R0LUDAJ.
Authentication URL: http://WINN6C3R0LUDAJ.
Enrollment URL: http://ipfvt-mylab.englab.brocade.com/CertSrv/mscep/mscep.dll
SCEP password: helloutthere
```

Enter the **show pki entity** command to display information on configured PKI entities.

```
device# show pki entity
----------------PKI ENTITY ENTRY-----------------
Entity Name: spatha27
Common Name: Spatha
Organization Name: SQA
Organization Unit Name: ICX
State Name: KA
Country Name: IN
----------------PKI ENTITY ENTRY-----------------
Entity Name: ent1
Common Name: en1
State Name: KA
Country Name: IN
Location: BLR
----------------PKI ENTITY ENTRY-----------------
Entity Name: entity1
Common Name: tester1
Organization Name: BRCD
Organization Unit Name: FI
State Name: BC
Country Name: CA
Email: user@brocade.com
Location: BG
```

HTTP and HTTPS

Web authentication overview

Authentication is important in enterprise networks because the network is considered a secure area: it contains sensitive data and a finite amount of resources. Unauthorized users must be prevented from accessing the network to protect the sensitive data and prevent the unnecessary consumption of resources.

The ideal authentication method blocks unauthorized users at the earliest possible opportunity. For internal enterprise networks, this can be controlled at the edge switch port. Two popular forms of port-based security authentication used at the edge switch are MAC authentication and 802.1x authentication. MAC authentication authenticates the MAC addresses of hosts or users that are attempting to access the network. This type of authentication requires no intervention from the host or user who is attempting to be authenticated. It is easy to use, but it can only authorize hosts; it cannot be used to authorize users. 802.1x authentication can authorize users or hosts. It is more flexible than the MAC authentication method; however, it requires more support, configuration, maintenance and user intervention than MAC authentication.

The Ruckus Web authentication method provides an ideal port-based authentication alternative to MAC authentication without the complexities and cost of 802.1x authentication. Users gain access to the network by opening a Web browser and entering a valid URL address using HTTP or HTTPS services. Instead of being routed to the URL, the user's browser is directed to an authentication Web page on the ICX switch, or an external authentication server (such as Ruckus Cloudpath, Aruba ClearPass, or Cisco ISE). The Web page prompts the user to enter a user ID and password or a passcode. The credentials a user enters are used by a trusted source to authenticate the user.

If the authentication is unsuccessful, the appropriate page is displayed on the host browser. The host is asked to try again or call for assistance, depending on what message is configured on the Web page. If the host MAC address is authenticated by the trusted source, a Web page is displayed with a hyperlink to the URL the host originally entered. If the user clicks on the link, a new window is opened and the the user is directed to the requested URL.

While a MAC address is in the authenticated state, the host can forward data through the ICX switch. The MAC address remains authenticated until one of the following events occurs:

- The host MAC address is removed from a list of MAC addresses that are automatically authenticated. (Refer to Specifying hosts that are permanently authenticated on page 308).
- The re-authentication timer expires and the host is required to re-authenticate (Refer to Configuring the re-authentication period on page 309).
- The host has remained inactive for a period of time and the inactive period timer has expired. (Refer to Forcing re-authentication after an inactive period on page 311.)
- All the ports on the VLAN on which Web Authentication has been configured are in a down state. All MAC addresses that are currently authenticated are de-authenticated (Refer to Forcing re-authentication when ports are down on page 311.)
- The authenticated client is cleared from the Web Authentication table. (Refer to Clearing authenticated hosts from the Web Authentication table on page 309).

The ICX switch can be configured to automatically authenticate a host MAC address. The host will not be required to login or re-authenticate (depending on the re-authentication period) once the MAC address passes authentication.

A host that is logged in and authenticated remains logged in indefinitely, unless a re-authentication period is configured. When the re-authentication period ends, the host is logged out. A host can log out at any time by pressing the Logout button in the Web Authentication Success page.

NOTE
The host can log out as long as the Logout window (Success page) is visible. If the window is accidentally closed, the host cannot log out unless the re-authentication period ends or the host is manually cleared from the Web Authentication table.

Captive portal authentication (external web authentication)

Captive portal authentication provides a means to authenticate clients through an external web server. A client that seeks web access to a network is redirected to the authentication web login page hosted on an external network access control (NAC) server (such as Ruckus Cloudpath, Aruba ClearPass, or Cisco ISE) that is integrated with the RADIUS server.

NOTE
Because the authentication server and web login page reside in an external server, captive portal authentication may also be referred to as external web authentication. These two terms are used interchangeably.

To equip the ICX switch to handle the HTTP redirection mechanism, configuration details specific to the NAC server such as virtual IP address, HTTP or HTTPS protocol port number, and login page details hosted on the NAC server must be specified on the switch. Upon receiving the redirected web access request, the NAC server honors the login page to the client which in turn submits the user login credentials. The NAC server reverts the credentials and sends the username, password, and default URL of the web page to the network-attached storage (NAS) or switch.

NOTE
For details on configuring external captive portal on a NAC server, refer to the user manual for the NAC server being used. ICX switches support Ruckus Cloudpath, Aruba ClearPass, and Cisco ISE servers.

- For the Ruckus Cloudpath server, refer to the *Cloudpath ES 5.2 Deployment Guide* (at this URL: https://support.ruckuswireless.com/documents/2006).
- For the Aruba ClearPass server, refer to the Aruba ClearPass Guest User Guide. Refer to the ClearPass Guest 6.4 User Guide, as the version used for validation is 6.4.
- For the Cisco ISE server, refer to Cisco Identity Services Engine documentation.

The ICX switch makes use of the credentials for initiating the authentication process through the RADIUS server, which is integrated with NAC server.

NOTE
The RADIUS server on the ICX switch and the one integrated with the NAC server must have the same configuration.

The RADIUS server validates the user credential information and, if the client is authenticated, the client is redirected to the URL provided by the server. For information about re-authentication and login failure behavior, refer to Configuring the re-authentication period on page 309 and Defining the Web Authentication cycle on page 309.

Captive Portal profile for external Web Authentication

The Captive Portal profile serves as a template that includes configuration details specific to the external server such as virtual IP address, HTTP or HTTPS protocol port number, and details of the login page hosted on the NAC server.

NOTE
The terms Captive Portal and external Web Authentication are used interchangeably.

The details configured in the Captive Portal profile enable the switch to handle the HTTP redirection mechanism and redirect the client to the login page hosted on the NAC server. The Captive Portal profile is then applied on an external Web Authentication-enabled VLAN.

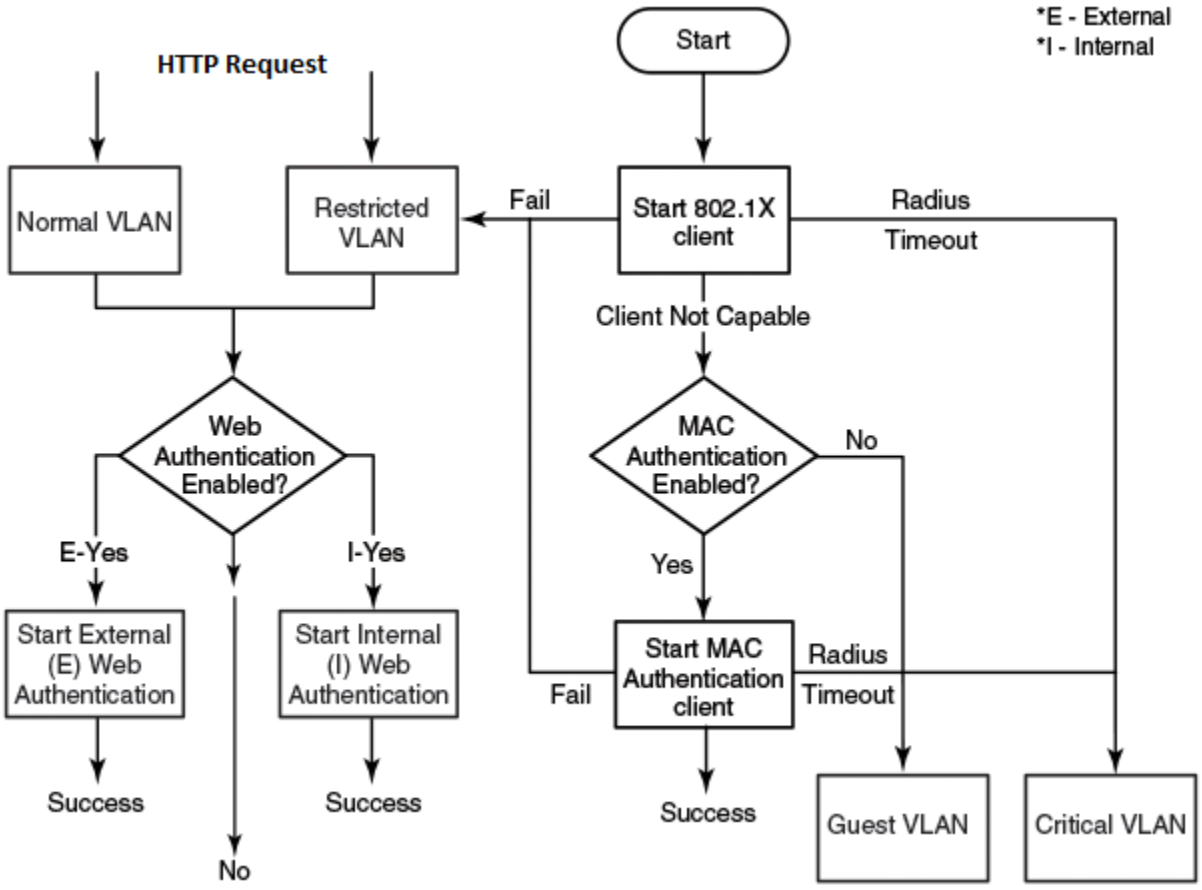
Captive Portal on a VLAN

A Captive Portal can be configured as a fallback authentication method for Flexible authentication (a combination of 802.1X authentication and MAC authentication).

A Captive Portal (sometimes referred to as External Web Authentication) can be enabled on any VLAN associated with Flexible authentication (restricted VLAN, guest VLAN, or critical VLAN). You can also enable a Captive Portal on any VLAN which is independent of Flexible authentication configuration. In either case, the client must send HTTP request for the switch to initiate external Web Authentication.

Figure 19 illustrates the Captive Portal flow on a restricted VLAN configured as part of Flexible authentication and a normal VLAN (any VLAN) which is independent of Flexible authentication configuration.

FIGURE 19 Captive Portal flow



Dynamic IP ACLs in Web Authentication

After successful authentication, different network policies can be applied to restrict the way the network resources are accessed by the client. Web Authentication implementation (internal and external) support dynamically applying an IP ACL to a port, based on information received from the authentication server.

When a client/supplicant is authenticated, the authentication server (the RADIUS server) sends the authenticator (the Ruckus 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 profile on the RADIUS server.

If the Access-Accept message contains the Filter-Id (type 11) attribute, the Ruckus device can use information in the attribute to apply an IP ACL filter to the authenticated port. The IP ACL filter applies to the port for as long as the client is connected to the network. The IP ACL filter is removed from the corresponding port when the client logs out.

NOTE
IPv6 ACL is not supported for Web Authentication.

The Ruckus device uses information in the Filter Id attributes as follows:

- The Filter-Id attribute can specify the number of an existing IP ACL filter configured on the Ruckus device. In this case, the IP ACL filter with the specified number is applied to the port.
- Dynamic ACLs are not supported in Layer 2 code when ACL per-port-per-VLAN is enabled.

After successful authentication, the RADIUS server may return an ACL that should be applied to the client on the port.

Configuration considerations for applying IP ACLs

- The name in the Filter-Id attribute is case-sensitive.
- IP ACLs must be extended ACLs. Standard ACLs are not supported.
- Dynamically assigned IP ACLs are subject to the same configuration restrictions as non-dynamically assigned IP ACLs.
- Filters are supported for inbound traffic only. Outbound filters are not supported.
- A maximum of one IP ACL per client can be configured in the inbound direction on an interface.
- Static ACLs are not supported with a Web Authentication-enabled port.
- Concurrent operation of a dynamic IP ACL and a static IP ACL is not supported.
- Dynamic IP ACL assignment with Web Authentication is not supported in conjunction with any of the following features:
  - IP Source Guard
  - Rate limiting
  - Protection against ICMP or TCP Denial of Service (DoS) attacks
  - Policy-based routing
  - DHCP snooping
  - ARP inspection
  - Flexible authentication dynamic IPv4 ACL and MAC filter
  - Static MAC filter
  - Static IPv4 access list
  - ACL logging

Dynamically applying existing ACLs

When a port is authenticated, an IP ACL filter that exists in the running configuration on the Ruckus device can be dynamically applied to the port. To do this, you must configure the Filter-Id (type 11) attribute on the RADIUS server. The Filter-Id attribute specifies the name or number of the Ruckus IP ACL filter.

The following table shows the standard RADIUS attribute as defined in RFC 2865 for IP ACL.

TABLE 34 Standard RADIUS attribute for the IP ACL

Attribute name	Attribute ID	Data type	Description
Filter-Id	11	String	IPv4 ACL ID or name as configured on the Ruckus device.

The following table shows the syntax for specifying the IP ACLs on the RADIUS server.

TABLE 35 Syntax for specifying the IP ACLs

Value	Description
ip.*number* .in	Applies the specified numbered IPv4 ACL to the port in the inbound direction.
ip.*name* .in	Applies the specified named IPv4 ACL to the port in the inbound direction.

RADIUS attribute for session timeout

Session timeout can be configured on the RADIUS server so that each client can have a different timeout value. The Session-Timeout attribute as defined in RFC 2865 is included in the Access-Accept message, and sets the maximum number of seconds of service to be provided to the user before termination of the session.

The following table shows the standard RADIUS attribute as defined in RFC 2865 for session timeout.

TABLE 36 Standard RADIUS attribute for session timeout

Attribute name	Attribute ID	Data type	Description
Session-Timeout	27	Integer	Session timeout after which session is cleared.

Web authentication configuration considerations

Web Authentication is modeled after other RADIUS-based authentication methods currently available on Ruckus edge switches. However, Web Authentication requires a Layer 3 protocol (TCP/IP) between the host and the authenticator. Therefore, to implement Web Authentication, you must consider the following configuration and topology configuration requirements:

- Web authentication works only when both the HTTP and HTTPS servers are enabled on the device.
- Web Authentication works only on the default HTTP or HTTPS port.
- The host must have an IP address prior to Web Authentication. This IP address can be configured statically on the host; however, DHCP addressing is also supported.
- If you are using DHCP addressing, a DHCP server must be in the same broadcast domain as the host. This DHCP server does not have to be physically connected to the switch. Also, DHCP assist from a router may be used.
- Web Authentication is not supported on a reserved VLAN.

The following applies to Web Authentication in the Layer 2 switch image:

- If the management VLAN and Web Authentication VLAN are in different IP networks, make sure there is at least one routing element in the network topology that can route between these IP networks.

The following are required for Web Authentication in the base Layer 3 and full Layer 3 images:

- Each Web Authentication VLAN must have a virtual interface (VE).
- The VE must have at least one assigned IPv4 address.

Web Authentication is enabled on a VLAN. That VLAN becomes a Web Authentication VLAN that does the following:

- Forwards traffic from authenticated hosts, just like a regular VLAN.
- Blocks traffic from unauthenticated hosts except from ARP, DHCP, DNS, HTTP, and HTTPs that are required to perform Web Authentication.

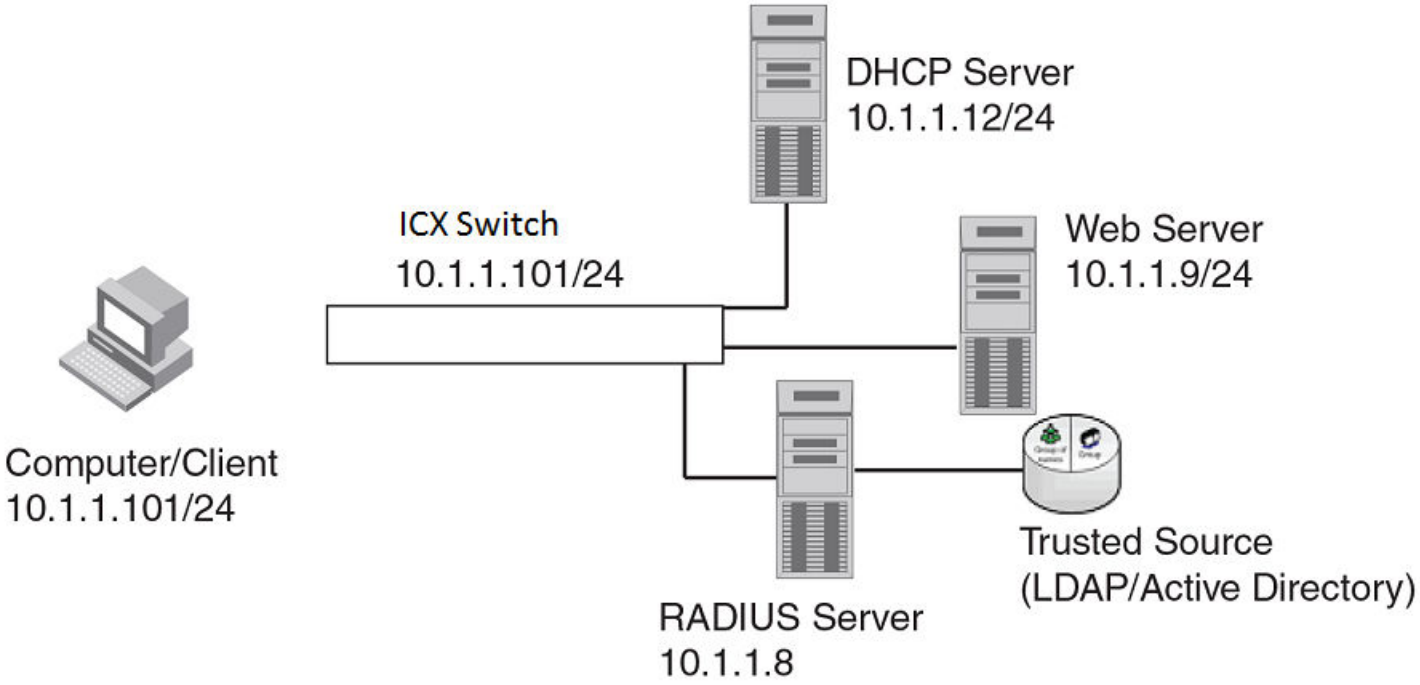
The *Basic topology for web authentication* figure shows the basic components of a network topology where Web Authentication is used. You will need:

- A Ruckus FastIron switch running a software release that supports Web Authentication
- DHCP server, if dynamic IP addressing is to be used
- Computer/host with a web browser

Your configuration may also require a RADIUS server with some Trusted Source such as LDAP or Active Directory.

NOTE
The Web server, RADIUS server, and DHCP server can all be the same server.

FIGURE 20 Basic topology for web authentication



Web Authentication configuration tasks

Complete the following steps to configure Web Authentication on a device.

1. Set up any global configuration required for the FastIron switch, RADIUS server, Web server and other servers.
   - On a Layer 2 FastIron switch, make sure the FastIron switch has an IP address.

     ```
     device# configure terminal
     device(config)# ip address 10.1.1.101/24
     ```

   - On a Layer 3 FastIron switch, assign an IP address to a virtual interface (VE) for each VLAN on which Web Authentication will be enabled.

     ```
     device#configure terminal
     device(config)# vlan 10
     device(config-vlan-10)# router-interface ve1
     device(config-vlan-10)# untagged e 1/1/1 to 1/1/10
     device(config-vlan-10)# interface ve1
     device(config-vif-1)# ip address 10.1.1.101/24
     ```

2. Configure the RADIUS server and other servers if Web Authentication will use a RADIUS server. By default, Web Authentication uses a RADIUS server to authenticate host usernames and passwords, unless it is configured to use a local user database.

```
device(config)# radius-server host 10.1.1.8 auth-port 1812 acct-port 1813 default key
2 $d3NpZ0BVXFpJ web-auth
```

NOTE
Remember the RADIUS key you entered. You will need this key when you configure your RADIUS server.

3. Configure Web Authentication to use secure (HTTPS) or non-secure (HTTP) login and logout pages. By default, HTTPS is used.

To enable the non-secure web server on the FastIron switch, enter the following commands.

```
device(config)# web-management HTTP
device(config# vlan 10
device(config-vlan-10# webauth
device(config-vlan-10-webauth# no secure-login
```

To enable the secure web server on the FastIron switch, enter the following commands.

```
device(config)# web-management HTTPS
device(config# vlan 10
device(config-vlan-10# webauth
device(config-vlan-10-webauth)# secure-login
```

4. Provide the switch with a certificate to enable Web Authentication using one of the following methods:

If the secure Web server is used, in order to access a secure Web page, the Web server needs to provide a key. This key is exchanged using a certificate. A certificate is a digital document that is issued by a trusted source that can validate the authenticity of the certificate and the Web server that is presenting it. Therefore the switch must have a certificate for web authentication to work.

- Upload a certificate using the following global configuration command.

```
device(config)# ip ssl private-key-file tftp ip-addr key-filename
```

- Generate a certificate using the following global configuration command.

```
device(config)# crypto-ssl certificate generate
```

5. Create a Web Authentication VLAN and enable Web Authentication on that VLAN.

```
device(config)# vlan 10
device(config-vlan-10)# webauth
device(config-vlan-10-webauth)# enable
```

When the Web Authentication is enabled, the CLI changes to the Web Authentication configuration mode. In the example, VLAN 10 requires hosts to be authenticated using Web Authentication before they can forward traffic.

6. Configure the Web Authentication mode:
   - Username and password: Blocks users from accessing the switch until they enter a valid username and password on a web login page.
   - Passcode: Blocks users from accessing the switch until they enter a valid passcode on a web login page.
   - captive-portal: Authenticates the users in a VLAN through external Web Authentication (Captive Portal user authentication) mode.
   - None: Blocks users from accessing the switch until they press the Login button. A username and password or passcode is not required.

   Refer to Web Authentication mode configuration on page 300.
7. Configure other Web Authentication options (refer to Web Authentication options configuration on page 307).

Prerequisites for Captive Portal support with Ruckus Cloudpath

The following are the prerequisites to support Captive Portal (external Web Authentication) on Ruckus ICX devices.

The parameters in the following table are mandatory while creating a guest or web login page on the Ruckus Cloudpath server.

For more details related to Web Logins page creation, refer to the *Cloudpath ES 5.2 Deployment Guide* (at this URL: https://support.ruckuswireless.com/documents/2006).

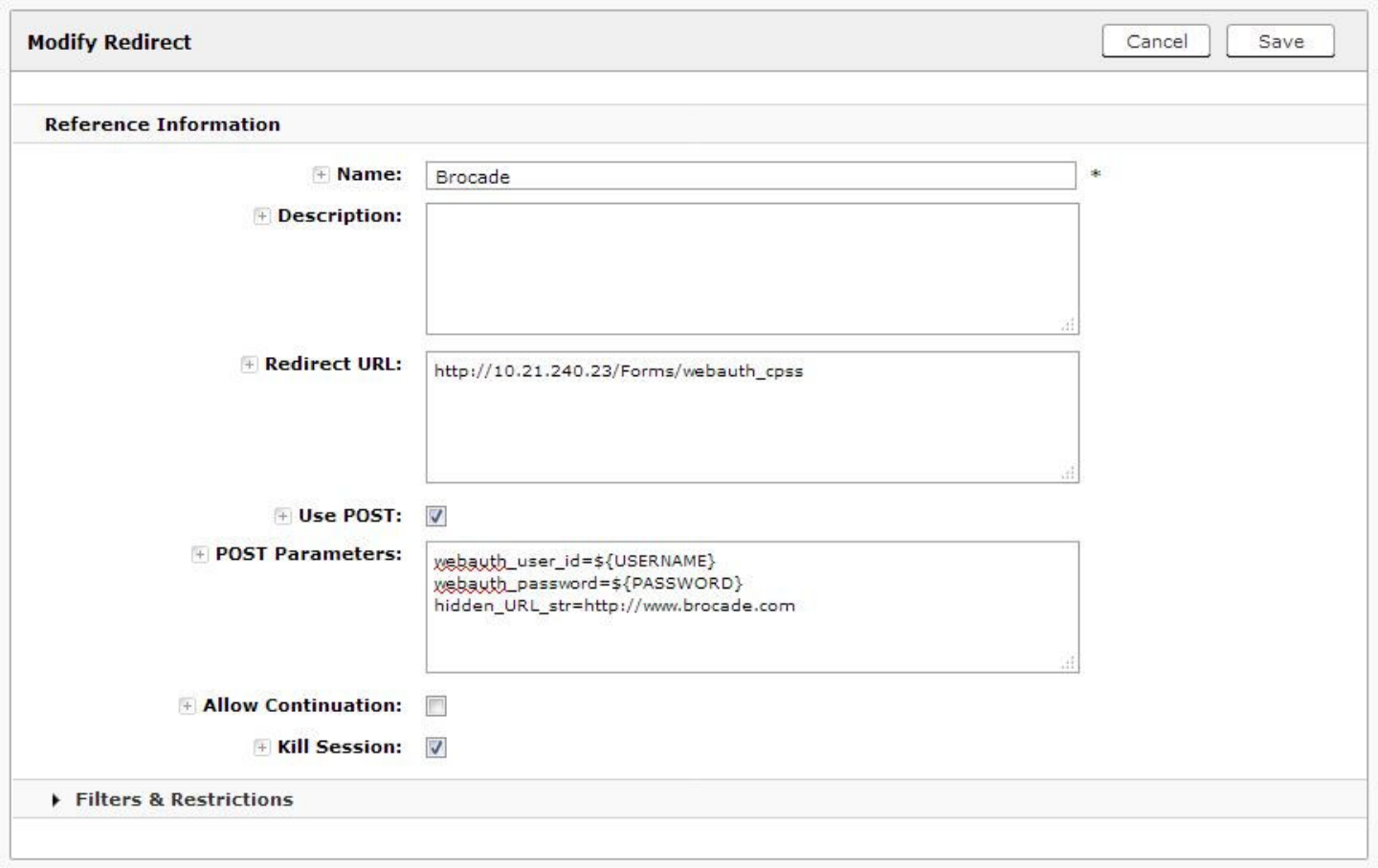
TABLE 37 Mandatory parameters to be added on the Ruckus Cloudpath server

Fields	Value	Description
Redirect URL	Use this syntax for a single ICX switch: http://<IP address>/Forms/webauth_cpss	Specifies the URL of the NAS device's login form.
Use POST	Must be selected.	Specifies the method to use while submitting the login form to NAS.
POST Parameters	webauth_user_id=${USERNAME} webauth_password=${PASSWORD} hidden_URL_str=http://www.ruckuswireless.com	Specifies the name of the username field for the login form, the name of the password field for the login form, and the destination field for the NAS device (the default URL value).

Other vendor-specific details are selected by default.

The following figure provides an example of the information required for Web Authentication Captive Portal Redirection.

FIGURE 21 Web Login configuration information



Prerequisites for configuring Captive Portal with Aruba ClearPass

The following are the prerequisites to support Captive Portal (external Web Authentication) on Ruckus ICX devices.

- Aruba ClearPass Policy Manager or CPPM for creating and managing the security profiles used for authentication.
- Aruba ClearPass Guest module for creating web logins pages for Guest access.

The parameters in the following table are mandatory while creating a guest or web login page on the Aruba ClearPass server.

For more details related to Web Logins page creation, refer to the "Configuration" section in the Aruba ClearPass Guest User Guide, release version 6.4.

TABLE 38 Mandatory parameters to be added on the Aruba ClearPass server

Fields	Value	Description
Submit URL	Use this syntax for a single ICX switch: http://<IP address>/Forms/webauth_cpss	Specifies the URL of the NAS device's login form.

TABLE 38 Mandatory parameters to be added on the Aruba ClearPass server (continued)

Fields	Value	Description
	Use this syntax for more than one ICX switches: {$switch_ip|default:"http://1.1.1.1/Forms/webauth_cpss"}	**NOTE** The **Submit URL** value for a single ICX switch differs from the **Submit URL** value for multiple ICX switches.
Submit Method	POST	Specifies the method to use while submitting the login form to NAS.
Username Field	webauth_user_id	Specifies the name of the username field for the login form. This is passed to the NAS device when the form is submitted.
Password Field	webauth_password	Specifies the name of the password field for the login form. This is passed to the NAS device when the form is submitted.
Extra Fields	url|hidden_URL_str!=	Use this field when original client requested URL needs to be re-directed
URL Field	hidden_URL_str	Specifies the destination field for the NAS device. This field contains the default URL value.
Default URL	Any URL Example https://www.ruckuswireless.com or http://www.ruckuswireless.com	Specifies the destination URL to which the client is redirected after authentication.

Other vendor-specific details are selected by default.

The following figures show examples of the information required for Web Authentication Captive Portal Redirection.

FIGURE 22 Web Login configuration information

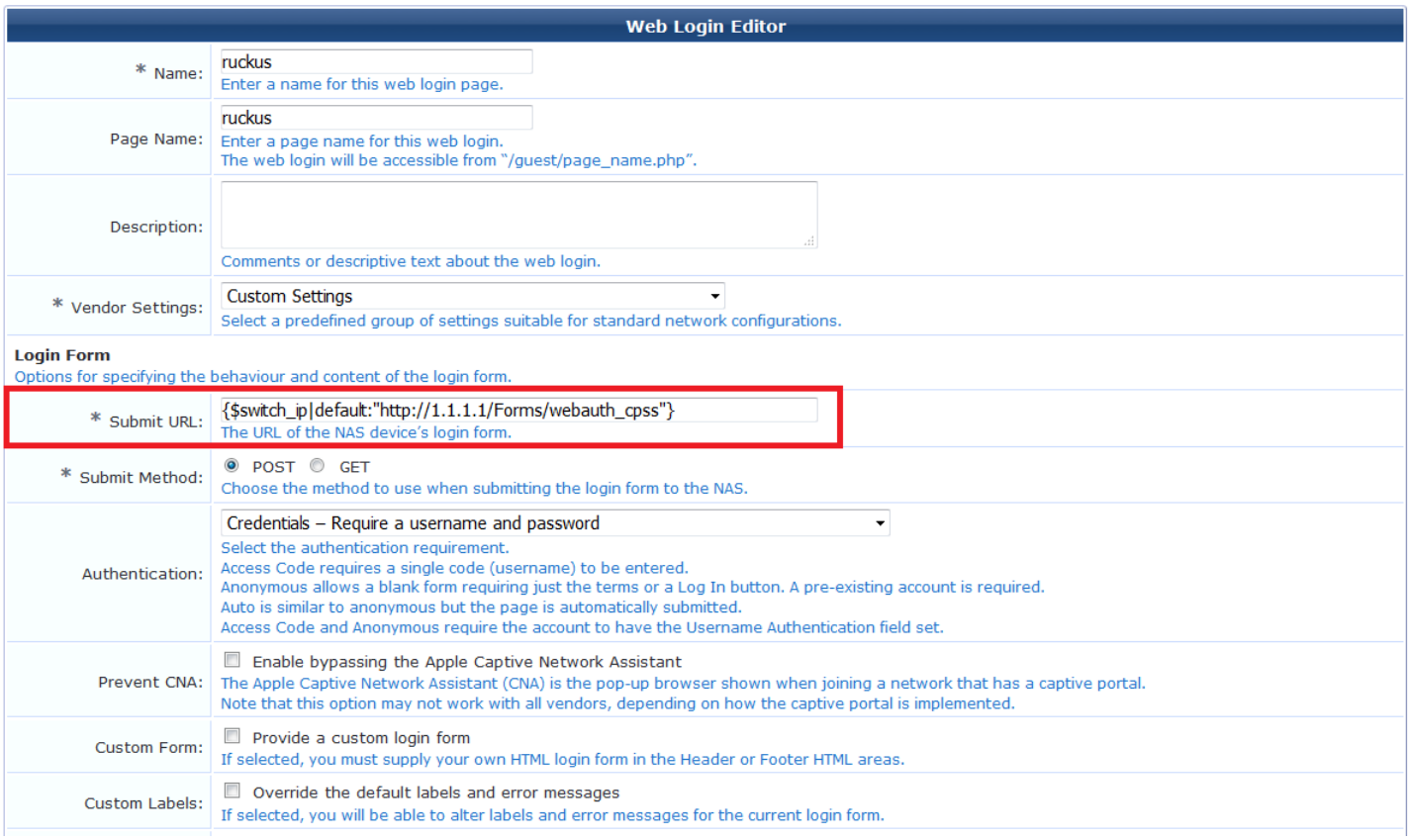
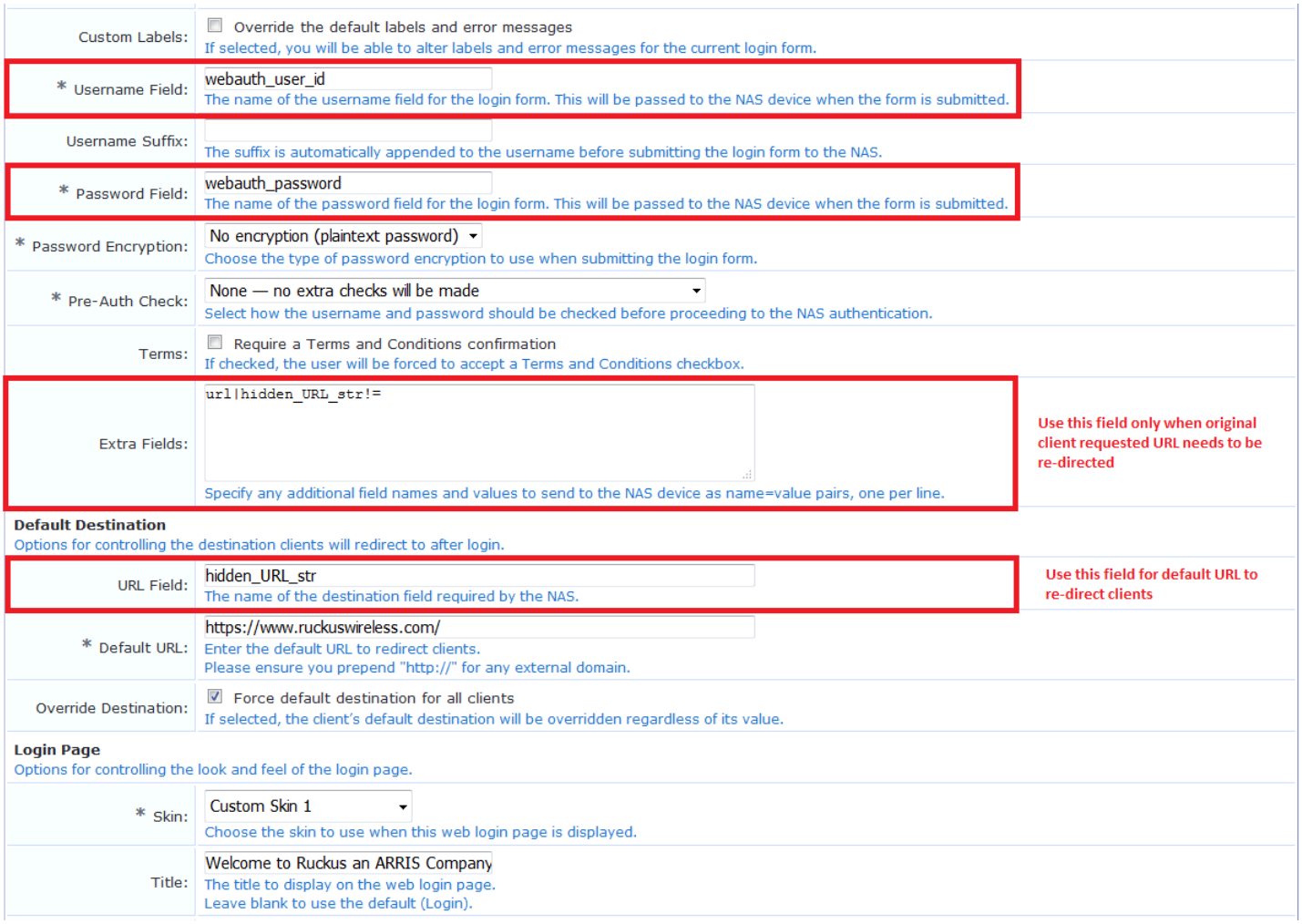


FIGURE 23 Web login configuration information (cont'd)



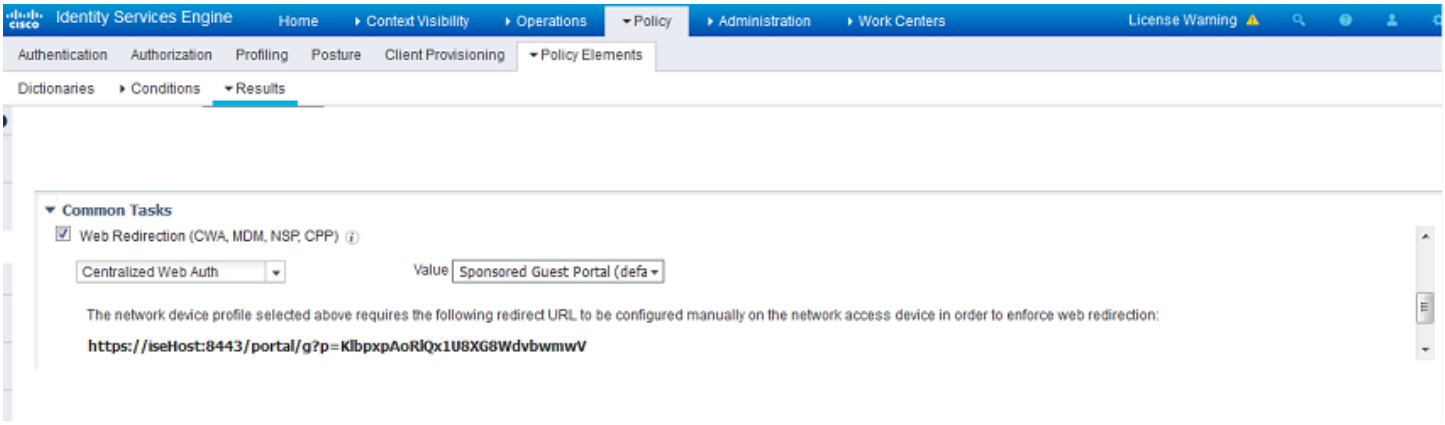
Prerequisites for configuring external Web Authentication with Cisco ISE

External Web Authentication (also called Captive Portal) on a Cisco ISE server is supported on Ruckus ICX devices.

Refer to the Cisco ISE server documentation for details of the redirect URL that must be configured on the Ruckus ICX devices. To enforce web redirection, you need to manually configure the redirect URL on the ICX devices.

The following figure shows the information required for Web Authentication Captive Portal Redirection.

FIGURE 24 Cisco ISE redirect URL



Prerequisite configurations on an ICX switch for Captive Portal authentication

The following are the prerequisites to support Captive Portal (external Web Authentication) on a Ruckus ICX device.

- Enable web management for HTTP and HTTPS access.
- Generate a crypto SSL certificate or import digital certificates issued by a third-party Certificate Authority (CA) to access a secure web page.
- Create Captive Portal profile that includes configuration details specific to the server such as virtual IP address, http or https protocol port number, and login-page details hosted on the NAC server.

Creating the Captive Portal profile for external Web Authentication

The following steps configure the Captive Portal profile for external Web Authentication:

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

   ```
   device# configure terminal
   ```

2. Enter the **captive-portal** command to create a user-defined Captive Portal profile.

   ```
   device(config)# captive-portal cp_ruckus
   ```

   The Captive Portal command mode is enabled, where you can specify the external policy server details that enable the switch to handle the HTTP redirection mechanism.

3. Enter the **virtual-ip** command to configure the IP address of the external policy server as the virtual IP address.

   You can enter the IP address or the DNS name, which will resolve to the IP address.

   ```
   device(config-cp-cp_ruckus)# virtual-ip 10.21.240.42
   ```

4. Enter the **virtual-port** command to configure the HTTP or HTTPS protocol port number to facilitate HTTP services for the clients in external Web Authentication.

   By default, HTTPS is used and the default port number for HTTPS is 443. You can also specify HTTP mode and the default port number for HTTP is 80.

   ```
   device(config-cp-cp_ruckus)# virtual-port 80
   ```

   The protocol configured in the Captive Portal profile must be the same as the protocol configured as part of web management access using the **web-management** command.
5. Enter the **login-page** command to configure the login page details to redirect the client to the login page hosted on the external policy server. Use one of the following options, depending on which NAC server you using:
   - if you are using Ruckus Cloudpath, use the following command: **login-page /enroll/***page-name*.

     ```
     device(config-cp-cp_ruckus)# login-page/enroll/ruckus/guestlogin
     ```

   - if you are using Aruba ClearPass, use the following command: **login-page /guest/***page-name*

     ```
     device(config-cp-cp_ruckus)# login-page/guest/ruckusguestlogin.php
     ```

   - if you are using Cisco ISE, use the following command: **login-page** *page-name created by the Cisco ISE server*.

     ```
     device(config-cp-cp_ruckus)# login-page ruckusguestlogin.php
     ```

   The login page details must be same as the login page hosted on the external policy server.
6. (Optional) Enter the **show captive-portal** command to view the output of the configured Captive Portal profile.

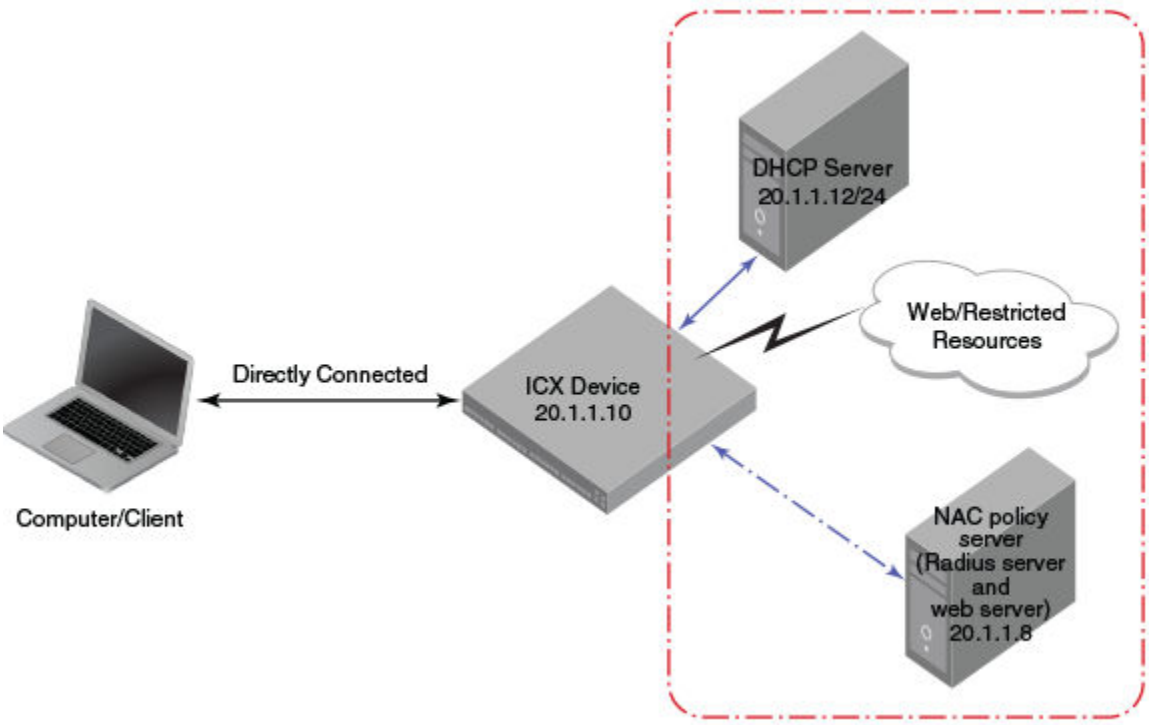
   In this example, the Captive Portal is configured with Ruckus Cloudpath as the NAC.

   ```
   device(config)# show captive-portal cp_ruckus
   Configured Captive Portal Profile Details :
     cp-name               :cp_ruckus
     virtual-ip            :10.21.240.42
     virtual-port          :80
     login-page            :/enroll/ruckus/guestlogin.php
   ```

Configuring Captive Portal (external Web Authentication)

On ICX switches web authentication is enabled at the VLAN level. A Captive Portal profile must be applied to the web authentication-enabled VLAN. For more information, refer to Creating the Captive Portal profile for external Web Authentication on page 296.

FIGURE 25 Basic network topology for a Captive Portal (external Web Authentication)



Complete the following steps to configure external Web Authentication on a device.

1. Set up any global configuration required for the ICX device, RADIUS server, NAC policy server, and other servers.
   - On a Layer 2 switch, make sure the ICX switch has an IP address configured.

     ```
     device# configure terminal
     device(config)# ip address 20.1.1.10/24
     ```

   - On a Layer 3 switch, assign an IP address to a virtual interface (VE) for each VLAN on which external Web Authentication will be enabled.

     ```
     device# configure terminal
     device(config)# vlan 20
     device(config-vlan-20)# router-interface ve20
     device(config-vlan-20)# untagged ethernet 1/1/1 to 1/1/20
     device(config-vlan-20)# interface ve20
     device(config-vif-20)# ip address 20.1.1.10/24
     ```

2. Configure the RADIUS server to authenticate the host username and passwords.

   The server has both a RADIUS server and a web server. Use the following commands to make RADIUS configuration on ICX switch.

   ```
   device(config)# radius-server host 20.1.1.8 auth-port 1812 acct-port 1813 default key
   2 $d3NpZ0BVXFpJ web-auth
   ```

   NOTE
   The RADIUS key configured should be the same as the key configured in the external web server.

3. Configure Web Authentication to use secure (HTTPS) or non-secure (HTTP) login and logout pages. By default, HTTPS is used.

   NOTE
   The protocol configured in the Captive Portal profile must be the same as the protocol configured as part of web management access.

   To enable the non-secure web server on the switch, enter the following commands.

   ```
   device(config)# web-management HTTP
   device(config)# vlan 20
   device(config-vlan-20)# webauth
   device(config-vlan-20-webauth)# no secure-login
   ```

   To enable the secure web server on the switch, enter the following commands.

   ```
   device(config)# web-management HTTPS
   device(config)# vlan 20
   device(config-vlan-20)# webauth
   device(config-vlan-20-webauth)# secure-login
   ```

4. Configure the key to access a secure web page using a certificate by performing one of the following steps:

   If the secure Web server is used, in order to access a secure web page, the web server needs to provide a key. This key is exchanged using a certificate. A certificate is a digital document that is issued by a trusted source that can validate the authenticity of the certificate and the web server that is presenting it. Therefore the switch must have a certificate for Web Authentication to work.

   - Upload a certificate using the following global configuration command.

     ```
     device(config)# ip ssl private-key-file tftp ip-addr key-filename
     ```

   - Generate a certificate using the following global configuration command.

     ```
     device(config)# crypto-ssl certificate generate
     ```

5. Create a Web Authentication VLAN and enable Web Authentication on that VLAN.

   ```
   device(config)# vlan 20
   device(config-vlan-20)# webauth
   device(config-vlan-20-webauth)# enable
   ```

   From this step onwards, the hosts must be authenticated to forward traffic.

6. Attach the configured Captive Portal profile to the Web Authentication-enabled VLAN.

   ```
   device(config-vlan-20-webauth)# captive-portal profile cp_ruckus
   ```

7. Configure the Web Authentication mode as Captive Portal mode to authenticate the users in a VLAN through external Web Authentication.

   ```
   device(config-vlan-20-webauth)# auth-mode captive-portal
   ```

8. Configure the external Captive Portal on the NAC server to create a guest or web login page for external Web Authentication.

Enabling and disabling Web Authentication

Web Authentication is disabled by default. To enable it, enter the following commands.

```
device(config)# vlan 10
device(config-vlan-10# webauth
device(config(config-vlan-10-webauth)# enable
```

The **vlan** command changes the CLI level to the VLAN configuration level. The **webauth** command changes the configuration level to the Web Authentication VLAN level. The **enable** command enables Web Authentication. In the example, VLAN 10 requires hosts to be authenticated using Web Authentication before they can forward traffic.

FastIron devices support a maximum of two Web Authentication VLANs.

Enter the **no enable** command to disable Web Authentication.

Web Authentication mode configuration

You can configure the FastIron device to use one of four Web Authentication modes:

- Username and password: Blocks users from accessing the device until they enter a valid username and password on a web login page. By default "username-password" is the authentication method. For more information, refer to Using local user databases on page 300.
- Passcode: Blocks users from accessing the device until they enter a valid passcode on a web login page. For more information, refer to Passcodes for user authentication on page 303.
- Captive Portal: Authenticates the users in a VLAN through external Web Authentication (Captive Portal user authentication) mode.
- None: Blocks users from accessing the device until they press the Login button. A username and password or passcode is not required. For more information, refer to Automatic authentication on page 307.

Using local user databases

Web Authentication supports the use of local user databases consisting of usernames and passwords, to authenticate devices. Users are blocked from accessing the switch until they enter a valid username and password on a web login page.

Once a user is authenticated successfully through username and password, the user is subjected to the same policies as for RADIUS-authenticated devices (for example, the re-authentication period, maximum number of users allowed, and so on). Similarly, once a user fails username and password authentication, the user is subjected to the same policies as for devices that fail RADIUS authentication.

You can create up to ten local user databases on the FastIron switch, either by entering a series of commands or by uploading a list of usernames and passwords from a TFTP file to the FastIron switch. The user databases are stored locally on the FastIron switch.

Configuring a local user database

The following steps configure a local user database.

1. Create the local user database.
2. Add records to the local user database, either by entering a series of commands or by importing a list of user records from an ASCII text file on the TFTP server to the FastIron switch.

3. Set the local user database authentication mode.
4. If desired, set the authentication method (RADIUS or local) failover sequence.
5. Assign a local user databse to a Web Authentication VLAN.

Creating a local user database

The FastIron switch supports a maximum of ten local user databases, each containing up to 50 user records. Each user record consists of a username and password.

To create a local user database, enter the **local-userdb** command.

```
device(config)# local-userdb userdb1
```

This example creates a local user database named userdb1. To add user records to this database, refer to Adding a user record to a local user database on page 301.

The local user database name can be up to 31 alphanumeric characters.

Adding a user record to a local user database

To add a user record, enter commands such as the following.

```
device(config)# local-userdb userdb1
device(config-localuserdb-userdb1)# username marcia password bunch4
```

The **local-userdb** command changes the configuration level to the local user database level. If the database does not already exist, it is created. The **username** command adds the user record to the database.

You can add up to 50 usernames and passwords to a local user database.

To view a list of users in a local user database, use the **show local-userdb** command. Refer to Displaying a list of local user databases on page 326.

Deleting a user record from a local user database

To delete a user record from the local user database, enter commands such as the following.

```
device(config)# local-userdb userdb1
device(config-localuserdb-userdb1)# no username marcia
```

The **local-userdb** command changes the configuration level to the local user database level . The **username** command deletes the user record from the database.

Deleting all user records from a local user database

To delete all user records from a local user database, enter the **delete-all** command.

```
device(config-localuserdb-userdb1)# delete-all
```

Creating a text file of user records

If desired, you can use TFTP to import a list of usernames and passwords from a text file on a TFTP server to the FastIron switch. The text file to be imported must be in the following ASCII format.

```
[delete-all]
[no] username
username1
```

```
 password
password1
 cr
[no] username
username2
 password
password2
 cr
...
```

The [delete-all] keyword indicates that the user records in the text file will replace the user records in the specified local user database on the FastIron switch. If the [delete-all] keyword is not present, the new user records will be added to the specified local user database on the FastIron switch. The [delete-all] keyword is optional. If present, it must appear on the first line, before the first user record in the text file.

The optional [no] keyword indicates that the user entry will be deleted from the specified local user database on the FastIron switch.

User records that already exist in the local user database will be updated with the information in the text file when it is uploaded to the switch.

Insert a cursor return (*cr*) after each user record.

You can enter up to 50 user records per text file.

Importing a text file of user records from a TFTP server

NOTE
Before importing the file, make sure it adheres to the ASCII text format described in Creating a text file of user records on page 301.

To import a text file of user records from a TFTP server to the FastIron switch, enter a command such as the following.

```
device(config-localuserdb-userdb1)# import-users tftp 192.168.1.1 filename userdb1
```

Using a RADIUS server as the Web Authentication method

By default, Web Authentication uses a RADIUS server to authenticate usernames and passwords of the hosts, unless the device is configured to use the local user database. You must perform the following steps.

1. Configure the RADIUS server information on the FastIron switch. Enter a command such as the following.

   ```
   device(config)# radius-server host 10.1.1.8 auth-port 1812 acct-port 1813 default key $GSig@U\
   ```

   NOTE
   Web Authentication uses the first reachable RADIUS server listed in the configuration. The **use-radius-server** command on individual ports is not supported for Web Authentication.

2. Enable the username and password authentication mode.

   ```
   device(config-vlan-10-webauth)# auth-mode username-password
   ```

3. Enable the RADIUS authentication method. Refer to Setting the local user database authentication method on page 303 or Setting the Web Authentication failover sequence on page 303

Setting the local user database authentication method

By default, the FastIron switch uses a RADIUS server to authenticate users in a VLAN. To configure the switch to use a local user database to authenticate users in a VLAN instead, enter the following command.

```
device(config-vlan-10-webauth)# auth-mode username-password auth-methods local
```

To revert back to using the RADIUS server, enter the following command.

```
device(config-vlan-10-webauth)# auth-mode username-password auth-methods radius
```

Setting the Web Authentication failover sequence

You can specify a failover sequence for the RADIUS and local user database authentication methods. For example, you can configure Web Authentication to first use a local user database to authenticate users in a VLAN. If the local user database is not available, it will use a RADIUS server. Enter the following command.

```
device(config-vlan-10-webauth)# auth-mode username-password auth-methods local radius
```

You can specify **radius local** or **local radius** depending on the failover sequence desired.

Assigning a local user database to a Web Authentication VLAN

After creating or importing a local user database on the FastIron switch and setting the local user database authentication method to **local**, you can configure a Web Authentication VLAN to use the database to authenticate users in a VLAN. To do so, enter a command such as the following.

```
device(config-vlan-10-webauth)# auth-mode username-password local-user-database userdb1
```

Use the **no** form of the command to remove the database from the Web Authentication VLAN.

Passcodes for user authentication

Web Authentication supports the use of passcodes to authenticate users. Users are blocked from accessing the switch until they enter a valid passcode on a web login page. Unlike username and password authentication, passcode authentication uses a simple number to authenticate users. The simplicity of a passcode reduces user errors and lowers the overhead of supporting and managing simple tasks, such as Internet access for guests and visitors in the office.

When passcodes are enabled, the system automatically generates them every 1440 minutes (24 hours), and when the system boots up. You can optionally create up to four static passcodes that will be used in conjunction with the dynamic passcodes generated by the system.

Configuring passcode authentication

The following steps configure the device to use the passcode authentication mode.

1. (Optional) Create up to four static passcodes.
2. Enable passcode authentication.
3. Configure other options.

Creating static passcodes

Static passcodes can be used for troubleshooting purposes, or for networks that want to use passcode authentication, but do not have the ability to support automatically-generated passcodes (for example, the network does not fully support the use of SNMP traps or Syslog messages with passcodes).

Manually-created passcodes are used in conjunction with dynamic passcodes . You can configure up to four static passcodes that never expire. Unlike dynamically created passcodes, static passcodes are saved to flash memory. By default, there are no static passcodes configured on the switch.

To create static passcodes, enter commands such as the following.

```
device(config-vlan-10-webauth)# auth-mode passcode static 3267345
device(config-vlan-10-webauth)# auth-mode passcode static 56127
```

The passcode can be a number from 4 to 16 digits in length. You can create up to four static passcodes, each with a different length. Static passcodes do not have to be the same length as passcodes that are automatically generated.

After creating static passcodes, enable passcode authentication.

To view the passcodes configured on the switch, use the **show webauth vlan** *vlan-id* **passcode** command. Refer to Displaying passcodes on page 326.

Enabling passcode authentication

To enable passcode authentication, enter the following command.

```
device(config-vlan-10-webauth)# auth-mode passcode
```

The **[no] auth-mode passcode** command enables Web Authentication to use dynamically created passcodes to authenticate users in the VLAN. If the configuration includes static passcodes, they are used in conjunction with dynamically created passcodes.

Enter **no auth-mode passcode** to disable passcode authentication.

Configuring the length of dynamically generated passcodes

By default, dynamically generated passcodes are 4 digits in length; for example, 0123. If desired, you can increase the passcode length to up to 16 digits. To do so, enter a command such as the following at the Web Authentication level of the CLI.

```
device(config-vlan-10-webauth)# auth-mode passcode length 10
```

The next dynamically created passcode will be 10 digits in length; for example, 0123456789.

The passcode can be a number from 4 to 16 digits in length.

Configuring the passcode refresh method

Passcode authentication supports two passcode refresh methods:

- Duration of time: By default, dynamically created passcodes are refreshed every 1440 minutes (24 hours). When refreshed, a new passcode is generated and the old passcode expires. You can increase or decrease the duration of time after which passcodes are refreshed, or you can configure the device to refresh passcodes at a certain time of day instead of after a duration of time.
- Time of day: When initially enabled, the time of day method will cause passcodes to be refreshed at 0:00 (12:00 midnight). If desired, you can change this time of day, and you can add up to 24 refresh periods in a 24-hour period.

When a passcode is refreshed, the old passcode will no longer work, unless a grace period is configured (refer to Configuring a grace period for an expired passcode on page 305).

If a user changes the passcode refresh value, the configuration is immediately applied to the current passcode. For example, if the passcode duration is 100 minutes and the passcode was last generated 60 minutes prior, a new passcode will be generated in 40 minutes. However, if the passcode duration is changed from 100 to 75 minutes, and the passcode was last generated 60 minutes prior, a new passcode will be generated in 15 minutes. Similarly, if the passcode duration is changed from 100 to 50 minutes, and the passcode was last generated 60 minutes prior, the passcode will immediately expire and a new passcode will be generated. The same principles apply to the time of day passcode refresh method.

If you configure both duration of time and time of day passcode refresh values, they are saved to the configuration file. You can switch back and forth between the passcode refresh methods, but only one method can be enabled at a time.

NOTE
Passcodes are not stateful, meaning a software reset or reload will cause the system to erase the passcode. When the FastIron switch comes back up, a new passcode will be generated.

Changing the passcode refresh duration

To change the duration of time after which passcodes are refreshed, enter a command such as the following.

```
device(config-vlan-10-webauth)# auth-mode passcode refresh-type duration 4320
```

The passcode will be refreshed after 4320 minutes (72 hours).

You can enter a value from 5 to 9999 minutes. The default is 1440 minutes (24 hours).

Refreshing passcodes at a certain time of the day

You can configure the FastIron switch to refresh passcodes at a certain time of day, up to 24 times each day, instead of after a duration of time. By default, passcodes will be refreshed at 00:00 (12:00 midnight).

To configure the switch to refresh passcodes at a certain time of day, enter commands such as the following.

```
device(config-vlan-10-webauth)# auth-mode passcode refresh-type time 6:00
device(config-vlan-10-webauth)# auth-mode passcode refresh-type time 14:30
```

The passcode will be refreshed at 6:00 am, 2:30 pm, and 12:00 midnight.

If you do not enter a passcode refresh time of day, by default, passcodes will be refreshed at 00:00 (12:00 midnight). You can configure up to 24 refresh times. Each must be at least five minutes apart.

Enter the **no** form of the command to remove the passcode refresh time of day.

Resetting the passcode refresh time of day configuration

If the FastIron switch is configured to refresh passcodes several times during the day, you can use the following comand to delete all of the configured times and revert back to the default time of 00:00 (12:00 midnight).

```
device(config-vlan-10-webauth)# auth-mode passcode refresh-type time delete-all
```

Configuring a grace period for an expired passcode

You can configure a grace period for an expired passcode. The grace period is the period of time that a passcode will remain valid, even after a new passcode is generated. For example, if a five-minute grace period is set and passcode 1234 is refreshed to 5678, both passcodes will be valid for five minutes. After the 1234 passcode expires, the 5678 passcode will remain in effect.

To configure the grace period for an expired passcode, enter a command such as the following.

```
device(config-vlan-10-webauth)# auth-mode passcode grace-period 5
```

The grace period can be from 0 through 5 minutes. Setting the grace period to 0 means there is no grace period.

NOTE
If the grace period is reconfigured while a passcode is already in the grace period, the passcode is not affected by the configuration change. The new grace period will apply only to passcodes that expire after the new grace period is set.

Flushing all expired passcodes that are in the grace period

You can delete old passcodes that have expired but are still valid because they are in the grace period. Flushing the expired passcodes is useful in situations where the old passcodes have been compromised but are still valid because of the grace period. Flushing the expired passcodes does not affect current valid passcodes or passcodes that newly expire.

To flush all expired passcodes that are currently in the grace period, enter the following command.

```
device(config-vlan-10-webauth)# auth-mode passcode flush-expired
```

Disabling and re-enabling passcode logging

A Syslog message and SNMP trap message are generated every time a new passcode is generated and passcode authentication is attempted,. This is the default behavior. If desired, you can disable passcode-related Syslog messages or SNMP trap messages, or both.

The following example shows a Syslog message and SNMP trap message related to passcode authentication.

```
New passcode: 01234567. Expires in 1440 minutes. Old passcode is valid for another 5 minutes.
```

To disable Syslog messages for passcodes, enter the **no auth-mode passcode log syslog** command.

```
device(config-vlan-10-webauth)# no auth-mode passcode log syslog
```

Enter the following command to disable SNMP trap messages for passcodes.

```
device(config-vlan-10-webauth)# no auth-mode passcode log snmp-trap
```

Enter the following command to re-enable Syslog messages for passcodes after they have been disabled.

```
device(config-vlan-10-webauth)# auth-mode passcode log syslog
```

Enter the following command to re-enable SNMP trap messages for passcodes after they have been disabled.

```
device(config-vlan-10-webauth)# auth-mode passcode log snmp-trap
```

Resending the passcode log message

If passcode logging is enabled, you can enter the **auth-mode passcode resend-log** command to retransmit the current passcode to a Syslog message or SNMP trap.

```
device(config-vlan-10-webauth)# auth-mode passcode resend-log
```

NOTE
The switch retransmits the current passcode only. Passcodes that are in the grace period are not sent.

Manually refreshing the passcode

You can manually refresh the passcode instead of waiting for the system to automatically generate one. When manually refreshed, the old passcode will no longer work, even if a grace period is configured. Also, if the passcode refresh duration of time method is used, the duration counter is reset when the passcode is manually refreshed. The passcode refresh time of day method is not affected when the passcode is manually refreshed.

To immediately refresh the passcode, enter the **auth-mode passcode generate** command.

```
device(config-vlan-10-webauth)# auth-mode passcode generate
```

Automatic authentication

By default, if Web Authentication is enabled, hosts must log in and enter authentication credentials in order to gain access to the network. If a re-authentication period is configured, the host will be asked to re-enter authentication credentials once the re-authentication period ends.

You can configure Web Authentication to authenticate a host when the user presses the Login button. When a host enters a valid URL address, Web Authentication checks the list of blocked MAC addresses. If the host's MAC address is not on the list and the number of allowable hosts has not been reached, after pressing the Login button, the host is automatically authenticated for the duration of the configured re-authentication period, if one is configured. Once the re-authentication period ends, the host is logged out and must enter the URL address again.

> **NOTE**
> Automatic authentication is not the same as permanent authentication. (Refer to Specifying hosts that are permanently authenticated on page 308). You must still specify devices that are to be permanently authenticated even if automatic authentication is enabled.

To enable automatic authentication, enter the following commands.

```
device(config)# vlan 10
device(config-vlan-10)# webauth
device(config-vlan-10-webauth)# auth-mode none
```

If automatic authentication is enabled and a host address is not in the blocked MAC address list, Web Authentication authenticates the host and displays the Login page without user credentials, and then provides a hyperlink to the requested URL site.

To determine if automatic authentication is enabled on your device, use the **show webauth vlan** command at the VLAN configuration level.

Syslog messages are generated under the following conditions:

- Automatic authentication is enabled.
- Automatic authentication is disabled.
- A MAC address is successfully authenticated.
- Automatic authentication cannot occur because the maximum number of hosts allowed has been reached.

Web Authentication options configuration

Web Authentication offers a number of other configuration options.

Enabling RADIUS accounting for Web Authentication

When Web Authentication is enabled, you can enable RADIUS accounting to record login (start) and logout (stop) events per host. The information is sent to a RADIUS server. Note that packet/byte count is not supported.

To enable RADIUS accounting, enter the **accounting** command.

```
device(config-vlan-10-webauth)# accounting
```

Enter the **no accounting** command to disable RADIUS accounting for Web Authentication.

Changing the login mode (HTTPS or HTTP)

Web Authentication can be configured to use secure (HTTPS) or non-secure (HTTP) login and logout pages. By default, HTTPS is used. Web Authentication pages on page 312 shows an example Login page.

To change the login mode to non-secure (HTTP), enter the **no secure-login** command.

```
device(config-vlan-10-webauth)# no secure-login
```

To revert to secure mode, enter the **secure-login** command.

```
device# secure-login
```

Specifying trusted ports

You can configure certain ports of a Web Authentication VLAN as trusted ports. All hosts connected to the trusted ports need not authenticate and are automatically allowed access to the network.

To create a list of trusted ports, enter commands such as the following.

```
device(config-vlan-10-webauth)# trust-port ethernet 1/1/3
device(config-vlan-10-webauth)# trust port ethernet 1/1/6 to 1/1/10
```

The command examples configure port 1/1/3 and ports 1/1/6 to 1/1/10 as trusted ports.

Specifying hosts that are permanently authenticated

Certain hosts, such as a DHCP server, gateways, and printers, may need to be permanently authenticated. Typically, these hosts are managed by the network administrator and are considered to be authorized hosts. Also, some of these hosts (such as printers) may not have a web browser and will not be able to perform Web Authentication.

To permanently authenticate these types of hosts, enter a command such as the following.

```
device(config-vlan-10-webauth)# add mac 0000.00eb.2d14 duration 0
```

The duration specifies how long the MAC address remains authenticated. The duration can be from 0 through 128000 seconds. The default is the current value of **reauth-time** . Setting the duration to 0 means that Web Authentication for the MAC address will not expire.

Instead of just entering a duration for how long the MAC address remains authenticated, you can specify the MAC address to be added by the specified port that is a member of the VLAN.

Enter the **no** form of the command to set the duration and ethernet to their default values. If you want to remove a host, enter the **no add mac** *mac-address* command.

NOTE
If a MAC address is statically configured, this MAC address will not be allowed to be dynamically configured on any port.

Configuring the re-authentication period

After a successful authentication, a user remains authenticated for a duration of time. At the end of this duration, the host is automatically logged off. The user must be re-authenticated again. To set the number of seconds a host remains authenticated before being logged off, enter a command such as the following.

```
device(config-vlan-10-webauth)# reauth-time 10
```

You can specify 0 through 128000 seconds. The default is 28800 seconds, and 0 means the user is always authenticated and will never have to re-authenticate, except if an inactive period less than the re-authentication period is configured on the Web Authentication VLAN. If this is the case, the user becomes de-authenticated if there is no activity and the timer for the inactive period expires.

Defining the Web Authentication cycle

You can set a limit as to how many seconds users have to be web-authenticated by defining a cycle time. This time begins at a user's first Login attempt on the Login page. If the user has not been authenticated successfully when this time expires, the user must enter a valid URL again to display the Web Authentication welcome page.

To define a cycle time, enter a command such as the following.

```
device(config-vlan-10-webauth)# cycle time 20
```

You can specify from 0 through 3600 seconds. The default is 600 seconds. Specifying 0 means there is no time limit.

Limiting the number of Web Authentication attempts

You can set a limit on the number of times a user enters an invalid username and password during the specified cycle time. If the user exceeds the limit, the user is blocked for a duration of time, which is defined by the **block duration** command. Also, the Web browser will be redirected to the exceeded allowable attempts web page.

To limit the number of Web Authentication attempts, enter a command such as the following.

```
device(config-vlan-10-webauth)# attempt-max-num 4
```

You can specify a number from 0 through 64. The default is 5. Specifying 0 means there is no limit to the number of Web Authentication attempts.

Clearing authenticated hosts from the Web Authentication table

You can clear dynamically authenticated hosts from the Web Authentication table.

To clear all authenticated hosts in a Web Authentication VLAN, enter a command such as the following.

```
device# clear webauth vlan 25 authenticated-mac
```

This command clears all the authenticated hosts in VLAN 25.

To clear a particular host in a Web Authentication VLAN, enter a command such as the following.

```
device# clear webauth vlan 25 authenticated-mac 0000.0022.3333
```

This command clears host 0000.0022.3333 from VLAN 25.

Setting and clearing the block duration for Web Authentication attempts

After users exceed the limit for Web Authentication attempts, you can specify how many seconds users must wait before the next cycle of Web Authentication begins. Enter the **block duration** command such as the following.

```
device(config-vlan-10-webauth)# block duration 4
```

Users cannot attempt Web Authentication during this time.

You can specify from 0 through 128000 seconds. The default is 90 seconds. Specifying 0 means that the MAC address is infinitely blocked.

To unblock the MAC address, wait until the block duration timer expires or enter a command such as the following.

```
device(config-vlan-10-webauth)# clear webauth vlan 10 block-mac 000.000.1234
```

If you do not specify a MAC address, then all the entries for the specified VLAN will be cleared.

Manually blocking and unblocking a specific host

A host can be temporarily or permanently blocked from attempting Web Authentication by entering a command such as the following.

```
device(config-vlan-10-webauth)# block mac 0000.00d1.0a3d duration 4
```

You can specify from 0 through 128000 seconds. The default is the current value of the **block duration** command. Specifying 0 means the MAC address is blocked permanently.

The **no block mac** *mac-address* **duration** *seconds* command resets duration to its default value.

You can unblock a host by entering the **no block mac** *mac-address* command.

Limiting the number of authenticated hosts

You can limit the number of hosts that are authenticated at any one time by entering a command such as the following.

```
device(config-vlan-10-webauth)# host-max-num 300
```

You can specify from 0 through 8192 hosts. The default is 0. Specifying 0 means there is no limit to the number of hosts that can be authenticated. The maximum of 8192 is the maximum number of MAC addresses the device supports.

When the maximum number of hosts has been reached, the FastIron switch redirects any new host that has been authenticated successfully to the Maximum Host web page.

Filtering DNS queries

Many of the Web Authentication solutions allow DNS queries to be forwarded from unauthenticated hosts. To eliminate the threat of forwarding DNS queries from unauthenticated hosts to unknown or untrusted servers (also known as domain-casting), you can restrict DNS queries from unauthenticated hosts to be forwarded explicitly to defined servers by defining DNS filters. Any DNS query from an unauthenticated host to a server that is not defined in a DNS filter is dropped. Only DNS queries from unauthenticated hosts are affected by DNS filters; authenticated hosts are not. If the DNS filters are not defined, then any DNS queries can be made to any server.

You can have up to four DNS filters. Create a filter by entering the following command.

```
device(config-vlan-10-webauth)# dns-filter 1 10.166.2.44/24
```

You can specify a number from 1 to 4 to identify the DNS filter.

You can specify the IP address and subnet mask of unauthenticated hosts that will be forwarded to the unknown or untrusted servers.

You can use a wildcard for the filter. The wildcard is in dotted-decimal notation (IP address) format. It is a four-part value, where each part is 8 bits (one byte) separated by dots, and each bit is a one or a zero. Each part is a number ranging from 0 through 255 (for example, 0.0.0.255). Zeros in the mask mean the packet source address must match the IP address. Ones mean any value matches.

Forcing re-authentication when ports are down

By default, the device checks the link state of all ports that are members of the Web Authentication VLAN and if the state of all the ports is down, then the device forces all authenticated hosts to re-authenticate. That is, the **port-down-authenticated-mac-cleanup** command that enforces re-authentication of all authenticated hosts when all the ports are down is enabled by default. However, hosts that were authenticated using the **add mac** command will remain authenticated; they are not affected by the **port-down-authenticated-mac-cleanup** command.

```
device(config-vlan-10-webauth)# port-down-authenticated-mac-cleanup
```

Forcing re-authentication after an inactive period

You can force Web Authentication hosts to be re-authenticated if they have been inactive for a period of time. The inactive duration is calculated by adding the **mac-age-time** that has been configured for the device and the configured **authenticated-mac-age-time**. (The **mac-age-time** command defines how long a port address remains active in the address table.) If the authenticated host is inactive for the sum of these two values, the host is forced to be re-authenticated.

To force authenticated hosts to re-authenticate after a period of inactivity, enter commands such as the following.

```
device(config)# mac-age-time 600
device(config)# vlan 23
device(config-vlan-23)# webauth
device(config-vlan-23-webauth)# reauth-time 303
device(config-vlan-23-webauth)# authenticated-mac-age-time 300
```

In the **authenticated-mac-age-time** command, you can specify a value from 0 through the value entered for the **reauth-time** command. The default is 3600.

Refer to "Changing the MAC age time and disabling MAC address learning" section in the *Ruckus FastIron Layer 2 Switching Configuration Guide* for details on the **mac-age-time** command. The default value for the **mac-age-time** command is 300 seconds and can be configured to be 0 or a value between 60 and 600 on the FastIron switch. If it is configured to be 0, then the MAC address does not age out due to inactivity.

Defining the Web Authorization redirect address

When a user enters a valid URL, the user is redirected to the switch Web Authentication page and the welcome page is displayed. By default, the Web Authentication address returned to the browser is the IP address of the FastIron switch. To prevent the display of error messages saying that the certificate does not match the name of the site, you can change this address so that it matches the name on the security certificates.

To change the address on a Layer 2 switch, enter a command such as the following at the global configuration level.

```
device(config)# webauth-redirect-address my.domain.net
```

To change the address on a Layer 3 switch, enter a command such as the following at the Web Authentication VLAN level.

```
device(config-vlan-10-webauth)# webauth-redirect-address my.domain.net
```

Entering "my.domain.net" redirects the browser to https://my.domain.net/ when the user enters a valid URL on the web browser.

You can enter any value up to 64 alphanumeric characters for the string, but entering the name on the security certificate prevents the display of error messages saying that the security certificate does not match the name of the site.

Deleting a Web Authentication VLAN

To delete a Web Authentication VLAN, enter the **[no] webauth** command.

```
device(config)# vlan 10
device(config-vlan-10)# no webauth
```

Web Authentication pages

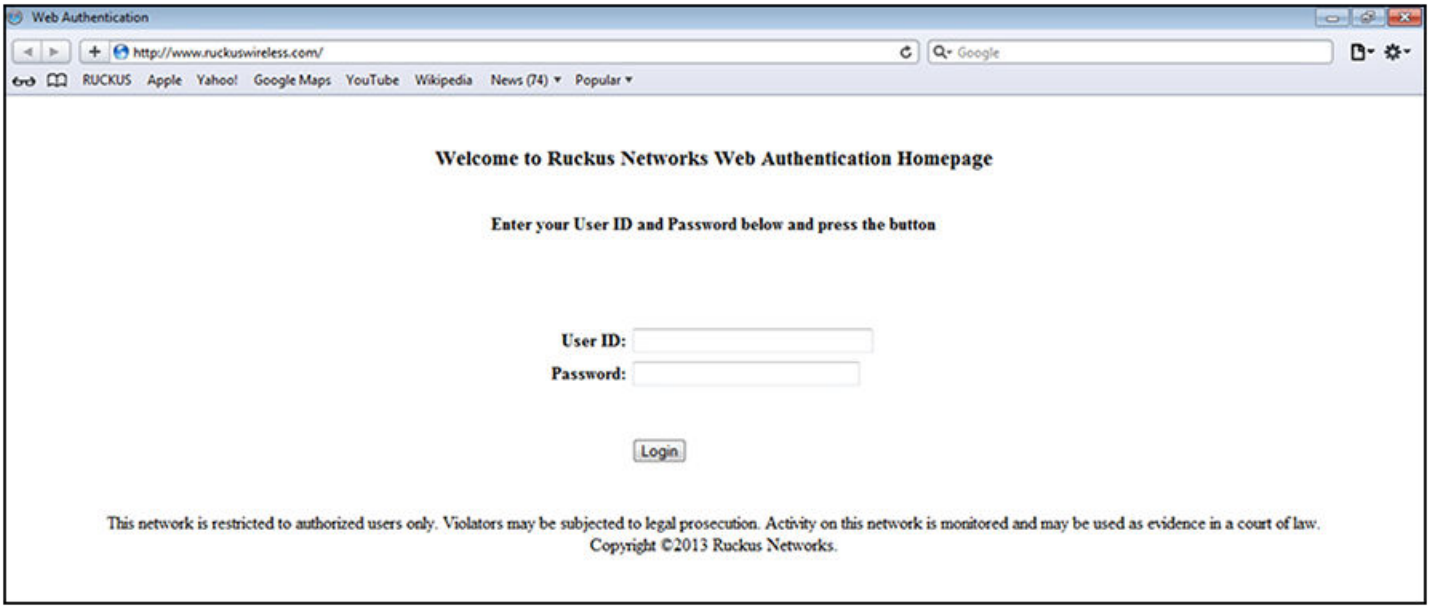
Several different web pages may be displayed during Web Authentication.

When a user enters a valid URL, the user is redirected to the switch Web Authentication page (refer to Defining the Web Authorization redirect address on page 312).

If automatic authentication is enabled, a welcome page appears. The browser will then be directed to the requested URL.

If username and password (local user database) authentication is enabled, the following login page appears.

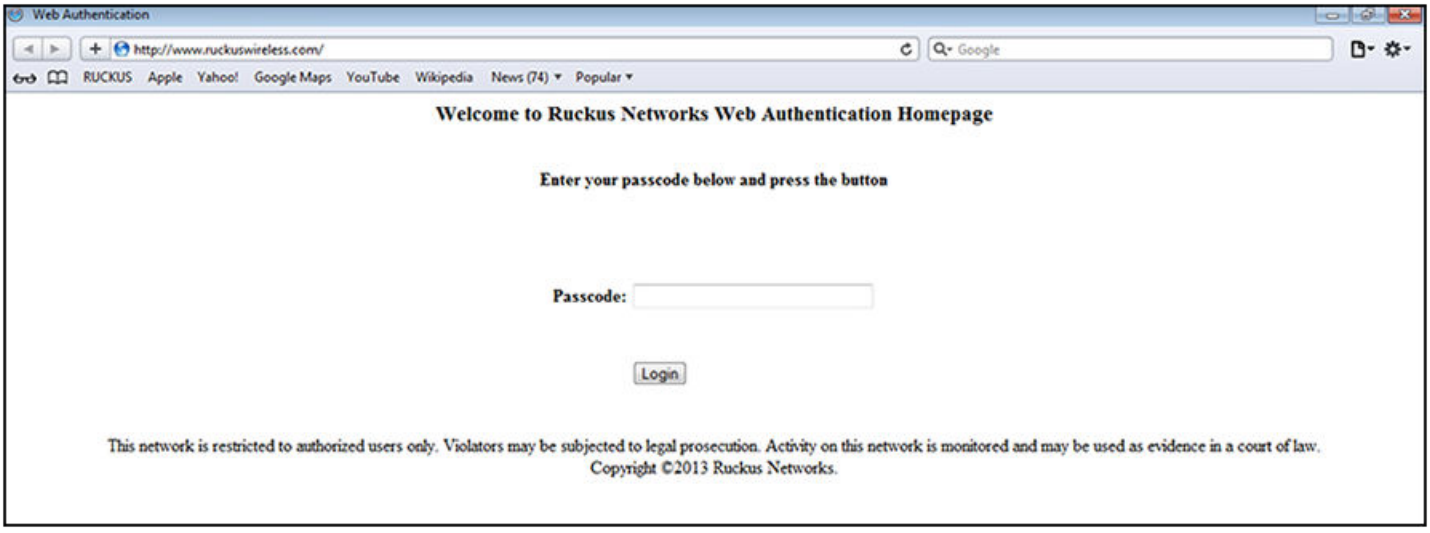
FIGURE 26 Example of login page when automatic authentication is disabled and a local user database is enabled



The user enters a username and password, which are sent for authentication.

If passcode authentication is enabled, the following login page appears.

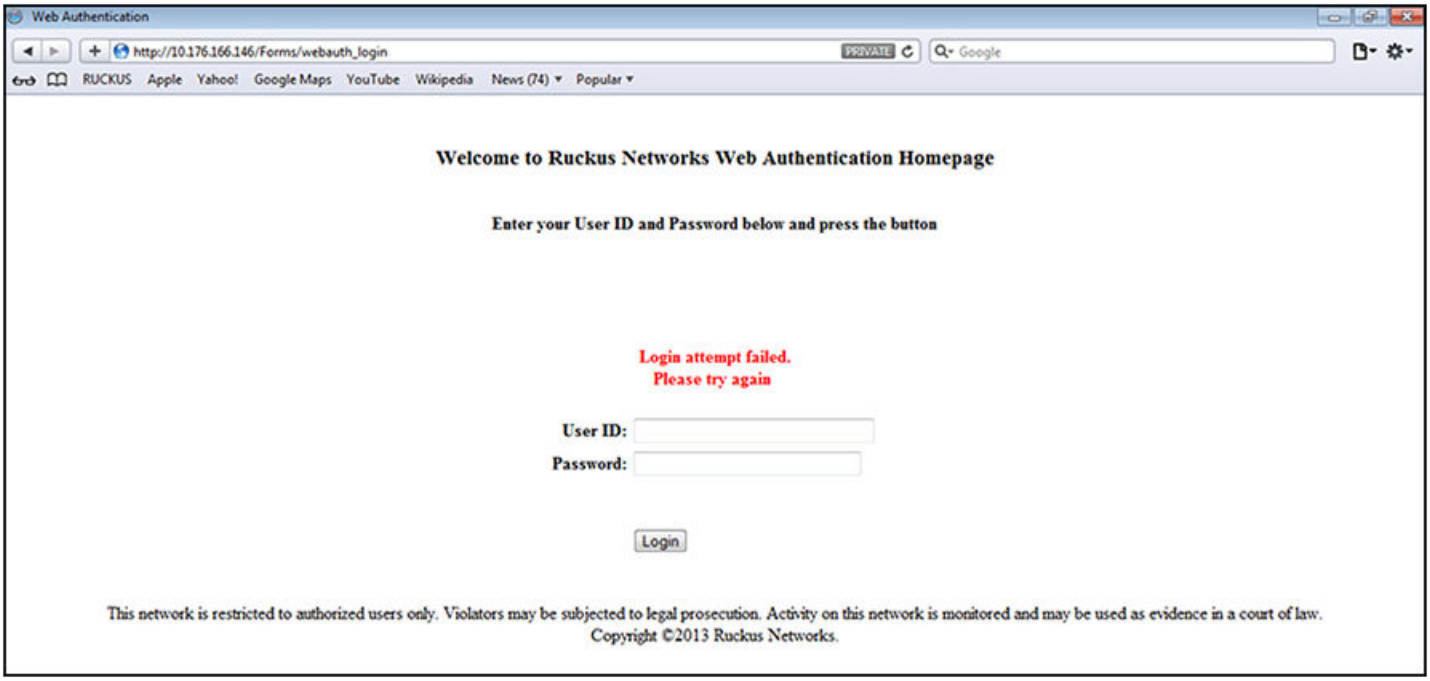
FIGURE 27 Example of login page when automatic authentication is disabled and passcode authentication is enabled



The user enters a passcode, which is sent for authentication.

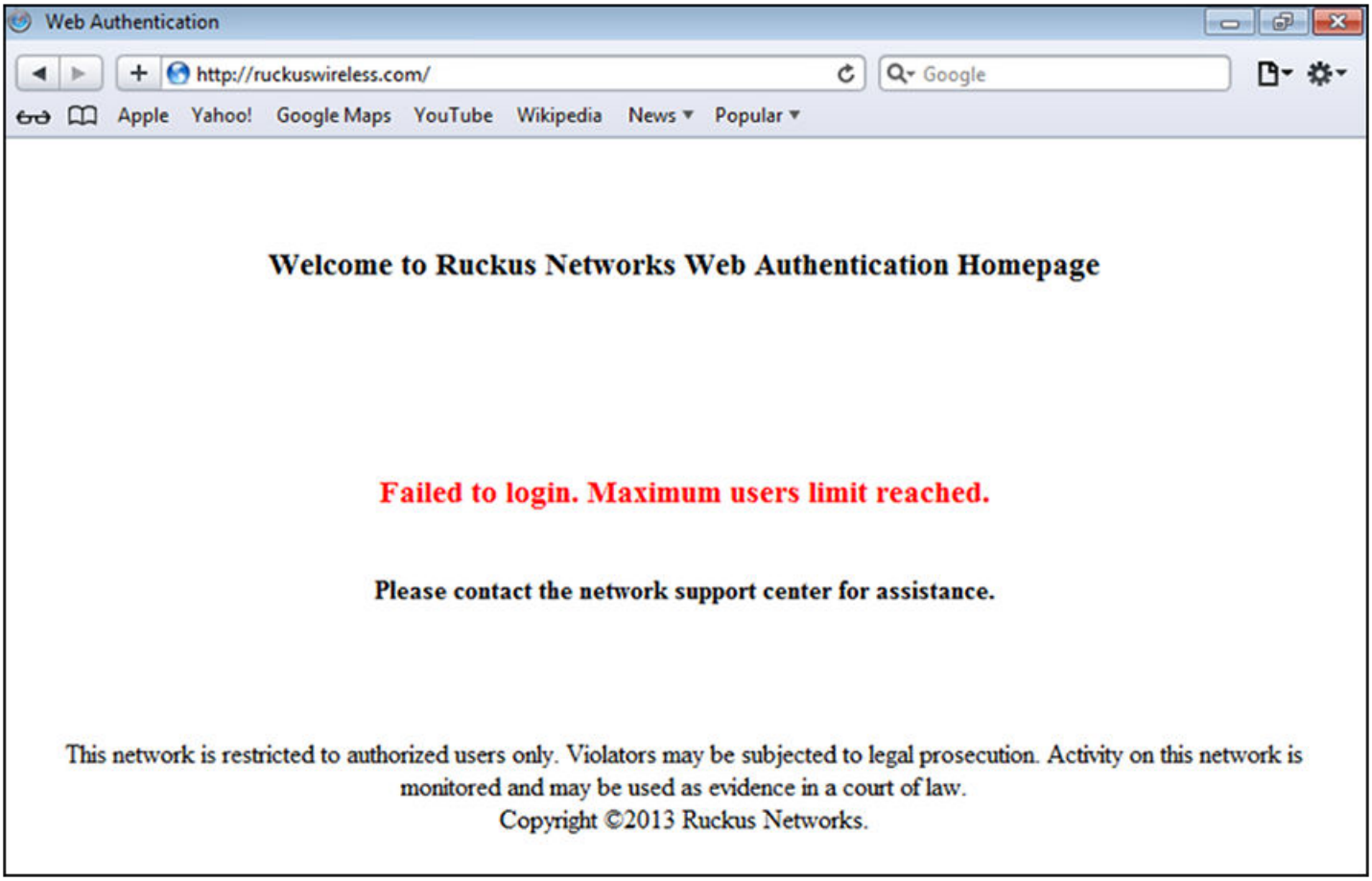
If Web Authentication fails, the following try again page appears.

FIGURE 28 Example of a try again page



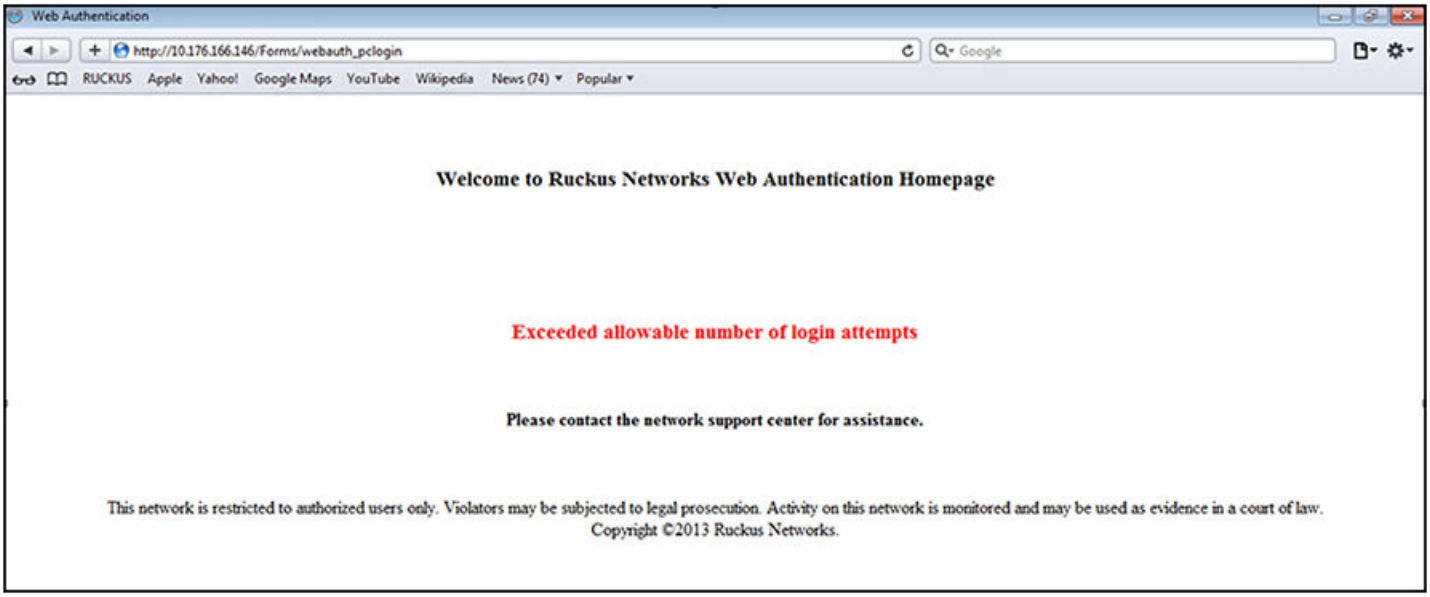
If the limit for the number of authenticated users on the network is exceeded, the following maximum host limit page appears.

FIGURE 29 Example of a maximum host limit page



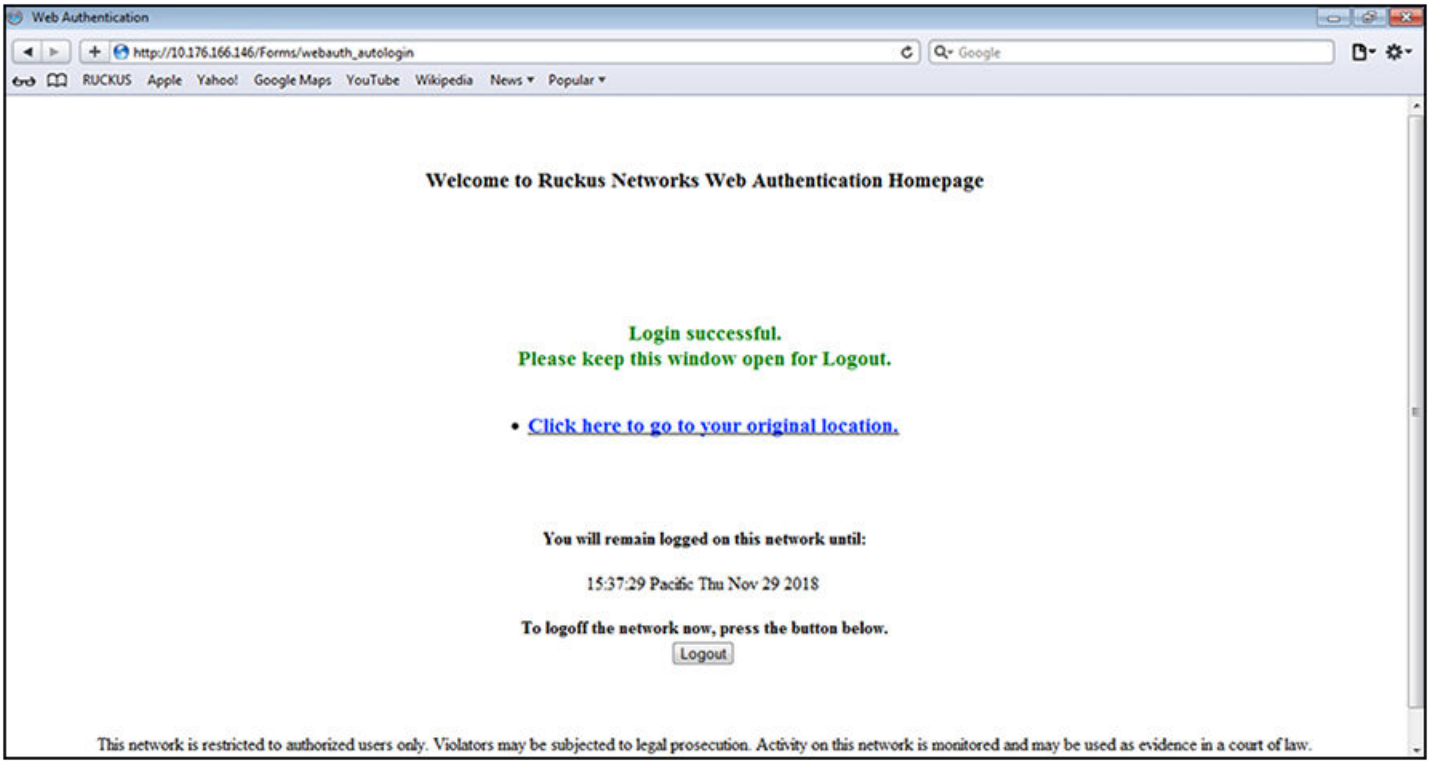
If the number of Web Authentication attempts by a user has been exceeded, the maximum attempts limit page is displayed. The user is blocked from attempting Web Authentication until either the user MAC address is removed from the blocked list (using the **clear webauth block-mac** command) or the block duration timer expires.

FIGURE 30 Example of a maximum attempts limit page



If Web Authentication is successful, the following success page appears.

FIGURE 31 Example of a Web Authentication success page

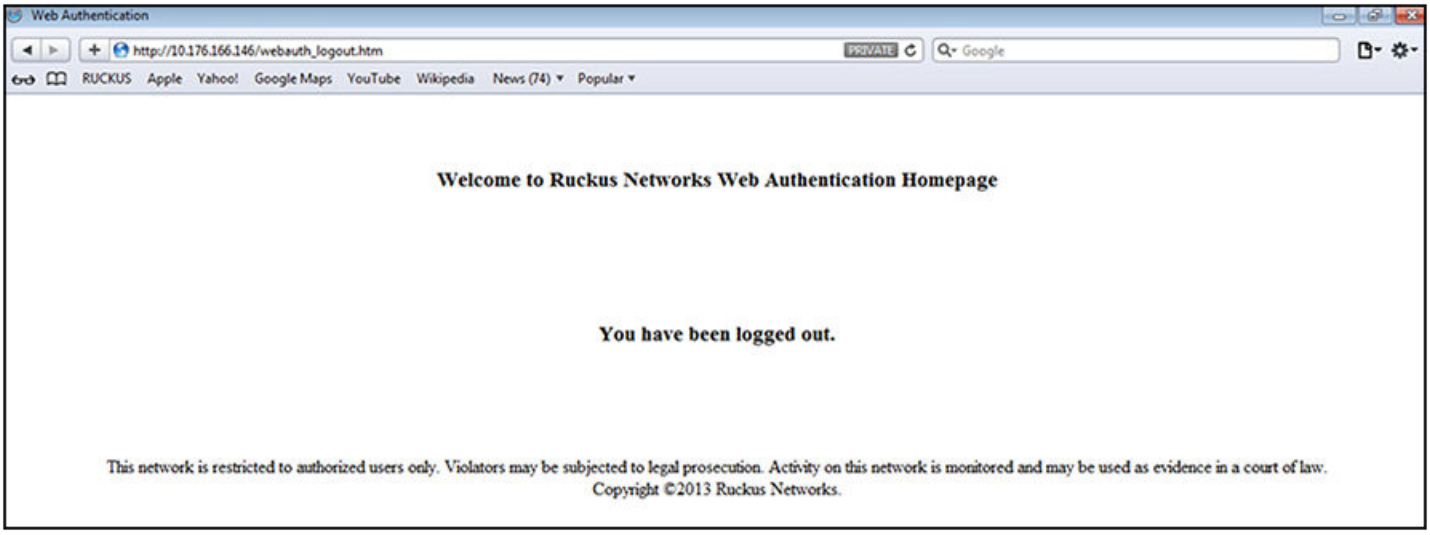


Once a host is authenticated, that host can manually de-authenticate by clicking the Logout button on the login success page. The host remains logged in until the re-authentication period expires. At that time, the host is automatically logged out. However, if a re-authentication period is not configured, the host remains logged in indefinitely.

NOTE
If you accidentally close the login success page, you will not be able to log out. If a re-authentication period is configured, you will be logged out once the re-authentication period ends.

The host can log out of the session by clicking the Logout button. Once logged out, the following window appears.

FIGURE 32 Example of a logout message page



You can customize the top and bottom text for the welcome page and all pages shown in the previous figures.

Displaying text for Web Authentication pages

Use the **show webauth vlan** *vlan-ID* **webpage** command to determine what text has been configured for Web Authentication pages.

```
device# show webauth vlan 25 webpage
==================================
Web Page Customizations (VLAN 25):
  Top (Header): Default Text
        "<h3>Welcome to Ruckus Wireless, Inc. Web Authentication Homepage</h3>"
  Bottom (Footer): Custom Text
        "Copyright 2018 SNL"
  Title: Default Text
        "Web Authentication"
  Login Button: Custom Text
        "Sign On"
  Web Page Logo:  blogo.gif
    align: left (Default)
  Web Page Terms and Conditions:  policy1.txt
```

Customizing Web Authentication pages

You can customize the following objects in the Web Authentication pages:

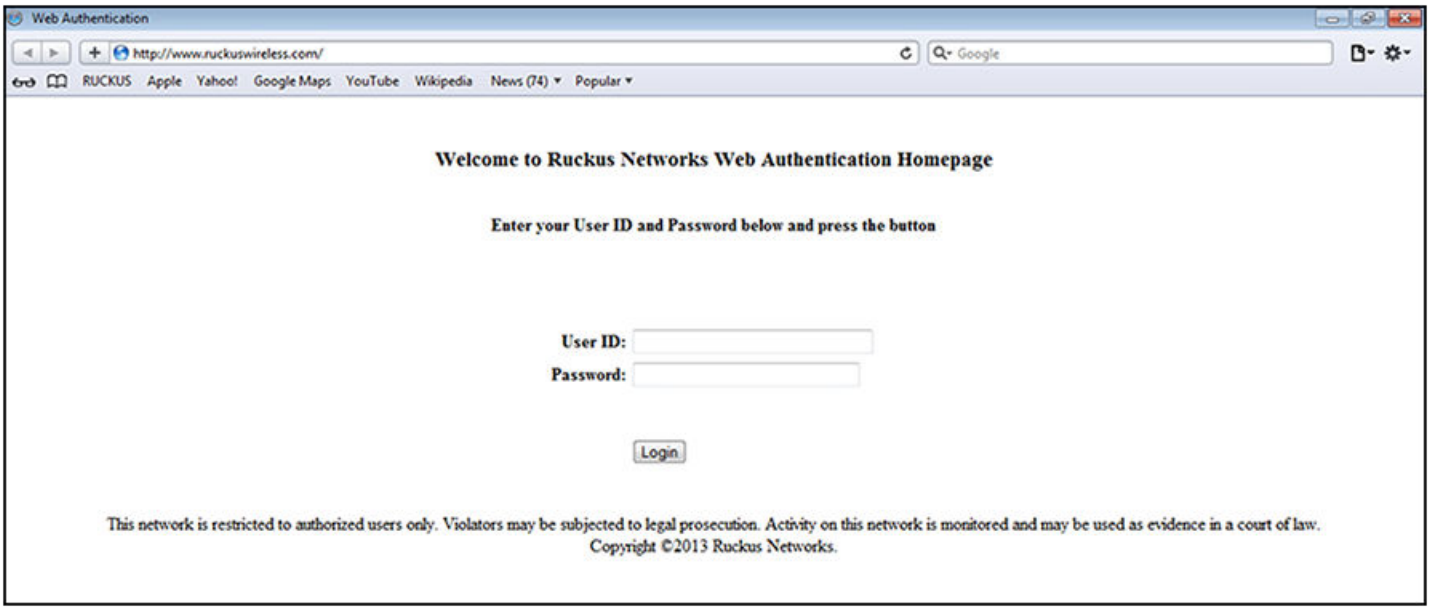
- Title bar
- Banner image (logo)
- Header
- Text box
- Login button
- Footer

You can use the **show webauth** and **show webauth vlan** *vlan-ID* **webpage** commands to determine what text has been configured for Web Authentication pages.

> **NOTE**
> The banner image does not apply to the Web Authentication maximum attempts limit page. The text box and Login button apply to the login page only.

The following figure shows the placement of these objects in the Login page.

FIGURE 33 Objects in the Web Authentication pages that can be customized



Customizing the title bar

You can customize the title bar that appears on all Web Authentication pages. To do so, enter a command such as the following.

```
device(config-vlan-10-webauth)# webpage custom-text title "Ruckus Secure Access Page"
```

You can enter up to 128 alphanumeric characters for the title bar. The default title bar is "Web Authentication".

To reset the title bar to the default value, enter the **no webpage custom-text title** command.

Customizing the banner image (logo)

You can customize the banner image (logo) that appears on all Web Authentication pages.

> **NOTE**
> The banner image does not display in the maximum attempts limit page.

To customize the banner image, use the TFTP protocol to upload an image file from a TFTP server to the FastIron switch. The image file can be in the .jpg, .bmp, or .gif format, and must be 64Kb or less. If you upload a new image file, it will overwrite the existing image file.

To replace the existing logo with a new one, enter a command such as the following.

```
device(config-vlan-10-webauth)# webpage logo copy tftp 10.10.5.1 ruckuslogo.gif
```

> **NOTE**
> The **webpage logo copy tftp** command downloads the image file and stores it in the device flash memory. Therefore, it is not necessary to follow this command with a **write memory** command.

Use the **no webpage logo** command to delete the logo from all Web Authentication pages and remove it from flash memory.

Aligning the banner image (logo)

You can configure the placement of the logo that appears on all Web Authentication pages. By default, the logo is left-aligned at the top of the page. To center the logo at the top of the page, enter the following command.

```
device(config-vlan-10-webauth)# webpage logo align center
```

To right-justify the log at the top of the page, enter the following command.

```
device(config-vlan-10-webauth)# webpage logo align right
```

Use the **no webpage logo align** command to reset the logo back to its default position.

Customizing the header

You can customize the header that appears on all Web Authentication pages.

To customize the header, enter a command such as the following.

```
device(config-vlan-10-webauth)# webpage custom-text top "Welcome to Network One"
```

You can enter up to 255 alphanumeric characters for the header. The default text is "Welcome to Ruckus Wireless, Inc. Web Authentication Homepage".

To reset the header back to the default text, enter the **no webpage custom-text top** command.

Customizing the text box

You can customize the text box that appears on the Web Authentication login page. By default, the text box is empty and is not visible. To create a text box or to replace the existing one, upload an ASCII text file from a TFTP server to the FastIron switch. The text file size must not exceed 2Kb.

To create or replace a text box, enter a command such as the following.

```
device(config-vlan-10-webauth)# webpage terms copy tftp 10.10.5.1 policy.txt
```

NOTE
The **webpage terms copy tftp** command downloads the text file and stores it in the device flash memory. Therefore, it is not necessary to follow this command with a **write memory** command.

To revert back to the default (no text box), enter the **no webpage terms** command.

Customizing the Login button

You can customize the Login button that appears on the bottom of the Web Authentication Login page. To do so, enter a command such as the following.

```
device(config-vlan-10-webauth)# webpage custom-text login-button "Press to Log In"
```

You can enter up to 32 alphanumeric characters for the Login button text. The default Login button text is "Login".

To reset the Login button to the default value, enter the **no webpage custom-text login-button** command.

Customizing the footer

You can customize the footer that appears on all Web Authentication pages.

To customize the footer, enter a command such as the following.

```
device(config-vlan-10-webauth)# webpage custom-text bottom "Network One Copyright 2010"
```

You can enter up to 255 alphanumeric characters for the footer text. The default text is "This network is restricted to authorized users only. Violators may be subjected to legal prosecution. Activity on this network is monitored and may be used as evidence in a court of law. Copyright 2018 Ruckus Wireless, Inc."

To reset the footer to the default text, enter the **no webpage custom-text bottom**.

Image Download over HTTPS

Support has been added for downloading images to flash memory over HTTPS. HTTPS image download involves downloading of an image from the remote HTTPS server to either the primary or secondary partition in a simple process.

After the **copy https flash** command is executed specifying the URL to the image file on the server (including IP address, port, path, and file name), an HTTP request is issued to download an image from the server. The response to the HTTP request contains the binary image in the body. After the image is downloaded by the HTTPS client, the data is written to a file in a specific location and image installation begins for either the primary or secondary partition.

Considerations for downloading images over HTTPS

- The flash memory is locked for the entire image download and installation process.
- Primary and secondary images are supported.
- The HTTPS client uses the device certificate when it is connected to the a SZ server. In other cases it can connect to a server, for example a Linux HTTPS server, without a device certificate.
- The remote end must serve the ICX image, either by a web server or have a web service providing the image for the URL specified in the HTTP request.
- No special HTTP headers are required.
- Image download over HTTPS is not available in FIPS or CC mode.

- Only IPv4 HTTPS download is supported.
- A successful HTTPS download provides a 200 response code.
- If a unified forwarding image (UFI) is specified, the UFI consists of the application image, the boot code image and the signature in one unified file. For more information on the UFI, refer to the *Software Upgrade and Downgrade* chapter in the *Ruckus FastIron Software Upgrade Guide*.

Downloading an image from a web server to flash memory

The following example copies the "SPR08070b1.bin" image from the HTTPS server to the flash primary partition. IP address 10.1.1.1 is specified and port 876 is specified.

```
device# copy https flash 10.1.1.1 SPR08070b1.bin primary port 876
```

The following example copies the "SPR08070b1.bin" image from the HTTPS server to the flash secondary partition. IP address 10.2.1.1 is specified. Because no port is specified, the default of 443 is used.

```
device# copy https flash 10.2.1.1 SPR08070b1.bin secondary
```

The following example copies a primary UFI image file from the HTTPS server to the flash primary partition. IP address 10.2.1.1 is specified and port 700 is specified. The UFI consists of the application image, the boot code image, and the FI signature in one unified file.

```
device# copy https flash 10.2.1.1 SPR08080b1ufi.bin primary port 700
```

Configuration Download over HTTPS

Support has been added for downloading a configuration file from the HTTPS server to the startup configuration file. HTTPS configuration download works in a simple process.

After the **https startup-config** command is executed specifying the URL to the file on the server (including IP address, port, path, and file name), an HTTP request is issued to download the configuration file from the server. The response to the HTTP request contains the configuration in the body. After the configuration is downloaded by the HTTPS client, the data returned is written to the startup configuration file, replacing the original startup configuration.

Considerations for downloading a configuration file over HTTPS

- The original startup configuration is overwritten with the new configuration file.
- A reboot is required for the new configuration to take effect.
- A syslog startup notification is generated automatically when the startup configuration is updated.
- The HTTPS client uses the device certificate.
- The remote end must serve the ICX startup, either by a web server or have a web service providing the configuration for the URL specified in the HTTP request.
- No special HTTP headers are required.
- Configuration download over HTTPS is not available in FIPS or CC mode.
- Only IPv4 HTTPS download is supported.

Downloading a configuration file from a web server to the startup configuration

The following example copies an ICX configuration from the HTTPS server, specifying IP address 10.2.1.1 and the “cfg/backup.cfg” file name. Port number 876 is also specified.

```
device# copy https startup-config 10.2.1.1 cfg/backup.cfg port 876
```

The following example copies an ICX configuration from the HTTPS server, specifying IP address 10.1.1.1 and the “cfg/backup.cfg” file name. Because no port is specified, the default of 443 is used.

```
device# copy https startup-config 10.1.1.1 cfg/backup.cfg
```

Configuration Upload over HTTPS

Support has been added for uploading of a copy of the running configuration file or the startup configuration file from a FastIron device to an HTTPS server. HTTPS configuration upload works in a simple process.

After the **copy startup-config https** or **copy running-config https** command is executed specifying the URL to be used to upload ether the startup configuration or the running configuration to a remote HTTPS server, either the running configuration is generated or the startup configuration is read from flash memory. An HTTP PUT request is issued to the URL of the server, along with the contents of the running or startup configuration in the body of the request. The URL is in the form of path/filename. The remote server handles this request. The application on the server receives the configuration and stores it, typically in a file or a database. The server responds with status code 204, indicating success.

Considerations for uploading a configuration file over HTTPS

- The remote end must have a web service to upload the data in the HTTPS request and store it remotely in a database or file system. The web service must accept the URL specified in the HTTPS request.
- The HTTPS client uses the device certificate.
- The Content-Type header is set to plain text.
- Configuration upload over HTTPS is not available in FIPS or CC mode.

Uploading the running configuration to a web server

The following example uploads a copy of the running configuration file from a device to the HTTPS server, and specifies port 200.

```
device# copy running-config https 10.1.1.1 upload/backup.cfg port 200
```

Uploading the startup configuration to a web server

The following example uploads a copy of the startup configuration file from a device to the HTTPS server. Because no port is specified, the default of 443 is used.

```
device# copy startup-config https 10.1.1.1 backup/icx.cfg
```

Displaying Web Authentication information

You can use a number of **show** commands to display information about Web Authentication.

Displaying the Web Authentication configuration

Enter the **show webauth** command to display the configuration for Web Authentication.

```
device# show webauth
==========================================================================
WEB AUTHENTICATION (VLAN 25): Enable
attempt-max-num: 5 (Default)
host-max-num: 0 (Default)
block duration: 90 (Default)
cycle-time: 600 (Default)
port-down-authenticated-mac-cleanup: Enable (Default)
reauth-time: 28800 (Default)
authenticated-mac-age-time: 3600 (Default)
dns-filter: Disable (Default)
authentication mode: username and password (Default)
  authentication methods: radius
    Local user database name:  <none>
Radius accounting: Enable (Default)
Trusted port list:  None
Secure Login (HTTPS): Enable (Default)
Web Page Customizations:
  Top (Header): Default Text
  Bottom (Footer): Custom Text
         "SNL Copyright 2009"
  Title: Default Text
  Login Button: Custom Text
         "Sign On"
  Web Page Logo:  blogo.gif
    align: left (Default)
  Web Page Terms and Conditions: policy1.txt
Host statistics:
 Number of hosts dynamically authenticated: 0
 Number of hosts statically authenticated: 2
 Number of hosts dynamically blocked: 0
 Number of hosts statically blocked: 0
 Number of hosts authenticating: 1
```

The **show webauth** command displays the following information.

TABLE 39 Field description of the show webauth command output

Field	Description
WEB AUTHENTICATION (VLAN #)	Identifies the VLAN on which Web Authentication is enabled.
attempt-max-num	The maximum number of Web Authentication attempts during a cycle.
host-max-num	The maximum number of users that can be authenticated at one time.
block duration	The number of seconds a user who failed Web Authentication must wait before attempting to be authenticated.
cycle-time	The number of seconds in one Web Authentication cycle.
port-down-authenticated-mac-cleanup	Whether this option is enabled or disabled. If enabled, all authenticated users are de-authenticated if all the ports in the VLAN go down.
reauth-time	The number of seconds an authenticated user remains authenticated. Once this timer expires, the user must re-authenticate.

TABLE 39 Field description of the show webauth command output (continued)

Field	Description
authenticated-mac-age-time	If a user is inactive, the number of seconds a user has before the user-associated MAC address is aged out. The user will be forced to re-authenticate.
dns-filter	Shows the definition of any DNS filter that has been set. (Refer to Filtering DNS queries on page 311.
authentication mode	The authentication mode: • username and password (default) • passcode • captive-portal • none Also displays configuration details for the authentication mode.
RADIUS accounting	Whether RADIUS accounting is enabled or disabled.
Trusted port list	The statically configured trusted ports of the Web Authentication VLAN.
Secure login (HTTPS)	Whether HTTPS is enabled or disabled.
Web Page Customizations	The current configuration for the text that appears on the Web Authentication pages. Either "Custom Text" or "Default Text" displays for each page type: • "Custom Text" means the message for the page has been customized. The custom text is also displayed. • "Default Text" means the default message that ships with the FastIron switch is used. The actual text on the Web Authentication pages can be displayed using the **show webauth vlan** *<vlan-id>* **webpage** command. Refer to Displaying text for Web Authentication pages on page 317.
Host statistics	The authentication status and the number of hosts in each state.

The **show webauth** command by itself displays information for all VLANs on which Web Authentication is enabled. The **show webauth vlan** *vlan-id vlan-id* command displays information for a specific VLAN.

Displaying a list of authenticated hosts

Enter the **show webauth allowed-list** command to display a list of hosts that are currently authenticated.

```
device# show webauth allowed-list
================================================================================
VLAN 3: Web Authentication, Mode: I = Internal E = External
--------------------------------------------------------------------------------
Web Authenticated List                Configuration   Authenticated Duration  Dynamic
MAC Address       User Name   mode    Static/Dynamic  HH:MM:SS                ACL
--------------------------------------------------------------------------------
000c.2973.a42b    ruckus      E       D               1 day, 11:33:16         acl1
1222.0a15.f045    super       E       D               1 day, 11:32:51         acl1
1222.0a15.f044    foundry     E       D               1 day, 11:32:48         acl1
1222.0a15.f043    ruckus      E       D               1 day, 11:32:47         acl1
1222.0a15.f042    spirent     E       D               1 day, 11:32:4          acl1
```

The **show webauth allowed-list** command displays the following information.

TABLE 40 Field description of the show webauth allowed-list command output

Field	Description
VLAN #: Web Authentication	The ID of the VLAN on which Web Authentication is enabled.
Mode	The client is authenticated using an internal server or external server.
Web Authenticated List MAC Address	The MAC addresses that have been authenticated.
User Name	The authenticated username.
Configuration Static/Dynamic	Whether the MAC address was dynamically (passed Web Authentication) or statically (added to the authenticated list using the **add mac** command) authenticated.
Authenticated Duration HH:MM:SS	The remainder of time the MAC address will remain authenticated.
Dynamic ACL	The dynamically assigned ACL.

Displaying a list of hosts attempting to authenticate

Enter the **show webauth authenticating-list** command to display a list of hosts that are trying to authenticate.

```
device# show webauth authenticating-list
================================================================================
VLAN 3: Web Authentication, AuthMode: I=Internal E=External
--------------------------------------------------------------------------------
Web Authenticating List             # of Failed  Cycle Time Remaining
MAC Address        User Name  Mode  Attempts     HH:MM:SS
--------------------------------------------------------------------------------
000c.2973.a42b     N/A        E     0            00:01:36
```

The **show webauth authenticating-list** command displays the following information.

TABLE 41 Field description of the show webauth authenticating-list command output

Field	Description
VLAN #: Web Authentication	The ID of the VLAN on which Web Authentication is enabled.
AuthMode	The client is authenticated using an internal server or external server.
MAC Address	The MAC addresses that are trying to be authenticated.
User Name	The User Name associated with the MAC address.
# of Failed Attempts	Number of authentication attempts that have failed.
Cycle Time Remaining	The remaining time the user has to be authenticated before the current authentication cycle expires. Once it expires, the user must enter a valid URL again to display the Web Authentication welcome page.

Displaying a list of blocked hosts

Enter the **show webauth blocked-list** command to display a list of hosts that are currently blocked from any Web Authentication attempt.

```
device# show webauth blocked-list
================================================================================
VLAN 3: Web Authentication, AuthMode: I=Internal E=External
--------------------------------------------------------------------------------
Block List                          Configuration       Block Duration Remaining
MAC Address     User Name  Mode     Static/Dynamic      HH:MM:SS
--------------------------------------------------------------------------------
000c.2973.a42b  User1      E        D                   00:00:04
```

The **show webauth blocked-list** command displays the following information.

TABLE 42 Field description of the show webauth blocked-list command output

Field	Description
VLAN #: Web Authentication	The ID of the VLAN on which Web Authentication is enabled.
AuthMode	The client is authenticated using an internal server or external server.
Web Block List MAC Address	The MAC addresses that have been blocked from Web Authentication.
User Name	The User Name associated with the MAC address.
Configuration Static/Dynamic	Whether the MAC address was dynamically or statically blocked. The **block mac** command statically blocks MAC addresses.
Block Duration Remaining	The remaining time the MAC address has before the user with that MAC address can attempt Web Authentication.

Displaying a list of local user databases

The **show local-userdb** command displays a list of all local user databases configured on the FastIron switch and the number of users in each database.

```
device# show local-userdb
================================================================================
Local User Database Name           : My_Database
Number of users in the database    : 4
================================================================================
Local User Database Name           : test
Number of users in the database    : 3
================================================================================
Local User Database Name           : test123
Number of users in the database    : 3
```

Displaying a list of users in a local user database

The **show local-userdb** command displays a list of all users in a particular local user database.

```
device# show local-userdb test
================================================================================
Local User Database : test
Username                        Password
--------                        --------
user1                           $e$&Z9'%*&+
user2                           $e$,)A=)65N,%-3*%1?@U
user3                           $e$5%&-5%YO&&A1%6%<@U
```

As shown in the example, passwords are encrypted in the command output.

Displaying passcodes

If the passcode Web Authentication mode is enabled, you can use the following command to display current passcodes.

```
device# show webauth vlan 25 passcode
Current Passcode : 1389
This passcode is valid for 35089 seconds
```

Displaying Captive Portal profile details

The **show captive-portal** command displays the details of the Captive Portal profile configured on the device.

```
device(config)# show captive-portal cp-ruckus
Configured Captive Portal Profile Details :
```

```
cp-name               :cp-ruckus
virtual-ip            :10.21.240.42
virtual-port          :80
login-page            :/enroll/ruckus/guestlogin
```

Protecting against Denial of Service Attacks

Denial of service protection overview

In a Denial-of-Service (DoS) attack, a router is flooded with useless packets for the purpose of slowing down or stopping normal operation.

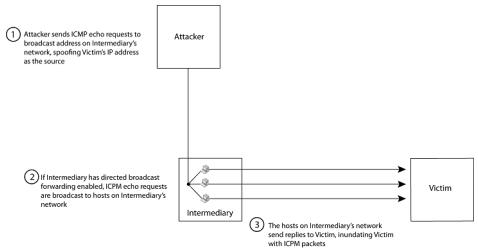
Ruckus ICX devices include measures to defend against two types of DoS attacks: Smurf attacks and TCP SYN attacks.

- Smurf attacks—Attacker sends ICMP echo request (ping) to broadcast address on the network of an intermediary and spoofs the IP address of the victim.
- TCP SYN attacks—Attacker floods a host with TCP SYN packets that have random source IP addresses that fill up the connection queue and service can be denied to legitimate TCP connections.

Protecting against smurf attacks

A smurf attack is a kind of DoS attack where an attacker causes a victim to be flooded with ICMP echo (pPing) replies sent from another network. Figure 34 illustrates how a smurf attack works.

FIGURE 34 How a smurf attack floods a victim with ICMP replies



The attacker sends an ICMP echo request packet to the broadcast address of an intermediary network. The ICMP echo request packet contains the spoofed address of a victim network as its source. When the ICMP echo request reaches the intermediary network, it is converted to a Layer 2 broadcast and sent to the hosts on the intermediary network. The hosts on the intermediary network then send ICMP replies to the victim network.

For each ICMP echo request packet sent by the attacker, a number of ICMP replies equal to the number of hosts on the intermediary network are sent to the victim. If the attacker generates a large volume of ICMP echo request packets, and the intermediary network contains a large number of hosts, the victim can be overwhelmed with ICMP replies.

Avoiding being an intermediary in a smurf attack

A smurf attack relies on the intermediary to broadcast ICMP echo request packets to hosts on a target subnet. When the ICMP echo request packet arrives at the target subnet, it is converted to a Layer 2 broadcast and sent to the connected hosts. This conversion takes place only when directed broadcast forwarding is enabled on the device.

To avoid being an intermediary in a smurf attack, make sure forwarding of directed broadcasts is disabled on the device. Directed broadcast forwarding is disabled by default. To disable directed broadcast forwarding, enter this command.

```
device(config)# no ip directed-broadcast
```

Avoiding being a victim in a smurf attack

You can configure the Ruckus device to drop ICMP packets when excessive numbers are encountered, as is the case when the device is the victim of a Smurf attack. You can set threshold values for ICMP packets that are targeted at the router itself or passing through an interface, and drop them when the thresholds are exceeded.

The number of incoming ICMP packets per second is measured and compared to the threshold values as follows:

- If the number of ICMP packets exceeds the **burst-normal** value, the excess ICMP packets are dropped.
- If the number of ICMP packets exceeds the **burst-max** value, all ICMP packets are dropped for the number of seconds specified by the **lockup** value. When the lockup period expires, the packet counter is reset and measurement is restarted.

  NOTE
  The burst-normal value parameter can be from 1 through 100,000 packets per second. The burst-max value parameter can be from 1 through 100,000 packets per second. The lockup seconds parameter can be from 1 through 10,000 seconds. This command is supported on Ethernet and Layer 3 interfaces.

Configuring threshold values for ICMP packets globally

Setting the threshold values for ICMP packets in global configuration mode.

1. Enter the global configuration mode to set the values globally.

   ```
   device# configure terminal
   ```

2. Set the threshold values for ICMP packets at the router.

   ```
   device(config)#ip icmp burst-normal 5000 burst-max 10000 lockup 300
   ```

   In the above example, if the number of ICMP packets received per second exceeds 5000, the excess packets are dropped. If the number of ICMP packets received per second exceeds 10,000, the device drops all ICMP packets for the next 300 seconds.

   NOTE
   The following example configures the threshold values for ICX 7750 device in global configuration mode. The "attack-rate" parameter is specific to ICX 7750 and has no associated value. For ICX 7750, the units of "burst-normal" and "burst-max" values are Kbps.

```
device# configure terminal
device(config)# ip icmp attack-rate burst-normal 2500 burst-max 3450 lockup 50
```

Configuring ICMP threshold values on an interface

Setting the threshold values for ICMP packets on interface configuration mode.

1. Enter the global configuration mode to set the values globally.

```
device# configure terminal
```

2. Specify the interface to be configured in the interface mode.

```
device(config)# interface ethernet 1/3/11
```

3. Specify the threshold values for ICMP packets that can be received per second.

```
device(config-if-e1000-1/3/11)# ip icmp burst-normal 5000 burst-max 10000 lockup 300
```

The following example set threshold values for ICMP packets received on interface 1/3/11 for a ICX 7750 device.

```
device(config)# configure terminal
device(config)# interface ethernet 1/3/11
device(config-if-e1000-1/3/11)# ip icmp attack-rate burst-normal 5000 burst-max 10000 lockup 300
```

For Layer 3 router code, if the interface is part of a VLAN that has a router VE, you must configure ICMP attack protection at the VE level. Otherwise, you can configure this feature at the interface level as shown above. When ICMP attack protection is configured at the VE level, it will apply to routed traffic only. It will not affect switched traffic.

You must configure VLAN information for the port before configuring ICMP attack protection. You cannot change the VLAN configuration for a port on which ICMP attack protection is enabled.

To set threshold values for ICMP packets received on virtual interface 31, enter the following commands.

```
device# configure terminal
device(config)# vlan 2
device(config-vlan-2)# router-interface ve 31
device(config-vlan-2)# interface ve 31
device(config-vif-31)# ip icmp burst-normal 5000 burst-max 10000 lockup 300
```

To set threshold values for ICMP packets received on virtual interface 31 for a ICX 7750 device, enter the following command in virtual interface mode.

NOTE
The attack-rate parameter is specific to ICX 7750 and has no associated value. For ICX 7750, the units of "burst-normal" and "burst-max" values are in Kbps.

```
device(config-vlan-2)# interface ve 31
device(config-vif-31)# ip icmp attack-rate burst-normal 5000 burst-max 10000 lockup 300
```

Protecting against TCP SYN attacks

TCP SYN attacks exploit the process of how TCP connections are established to disrupt normal traffic flow. When a TCP connection starts, the connecting host first sends a TCP SYN packet to the destination host. The destination host responds with a SYN ACK packet, and the connecting host sends back an ACK packet. This process, known as a "TCP three-way handshake," establishes the TCP connection.

While waiting for the connecting host to send an ACK packet, the destination host keeps track of the as-yet incomplete TCP connection in a connection queue. When the ACK packet is received, information about the connection is removed from the connection queue. Usually there is not much time between the destination host sending a SYN ACK packet and the source host sending an ACK packet, so the connection queue clears quickly.

In a TCP SYN attack, an attacker floods a host with TCP SYN packets that have random source IP addresses. For each of these TCP SYN packets, the destination host responds with a SYN ACK packet and adds information to the connection queue. However, because the source host does not exist, no ACK packet is sent back to the destination host, and an entry remains in the connection queue until it ages out (after approximately a minute). If the attacker sends enough TCP SYN packets, the connection queue can fill up, and service can be denied to legitimate TCP connections.

To protect against TCP SYN attacks, you can configure the Ruckus device to drop TCP SYN packets when excessive numbers are encountered. You can set threshold values for TCP SYN packets that are targeted at the router itself or passing through an interface, and drop them when the thresholds are exceeded.

The number of incoming TCP SYN packets per second is measured and compared to the threshold values as follows:

- If the number of TCP SYN packets exceeds the **burst-normal** value, the excess TCP SYN packets are dropped.
- If the number of TCP SYN packets exceeds the **burst-max** value, all TCP SYN packets are dropped for the number of seconds specified by the **lockup** value. When the lockup period expires, the packet counter is reset and measurement is restarted.

Configuring threshold values for TCP SYN packets globally

Setting threshold values for TCP SYN packets will protect against TCP SYN attack.

NOTE
The burst-normal value can be from 1 - 100,000 packets per second. The burst-max value can be from 1 - 100,000 packets per second. The lockup seconds can be from 1 - 10,000 seconds.

1. Enter the global configuration mode to set the values globally.

```
device# configure terminal
```

2. Set the threshold values for TCP SYN packets at the router.

```
device(config)# ip tcp burst-normal 10 burst-max 100 lockup 300
```

In the example, if the number of TCP SYN packets received per second exceeds 10, the excess packets are dropped. If the number of TCP SYN packets received per second exceeds 100, the device drops all TCP SYN packets for the next 300 seconds.

```
device(config)# configure terminal
device(config)# ip tcp burst-normal 10 burst-max 100 lockup 300
```

Configuring TCP SYN threshold values on an interface

Setting threshold values for TCP SYN packets received on an interface.

For ICX 7750 devices, the "attack rate" parameter is only applicable for smurf attacks and not for TCP/SYN attacks.

This command is available at the global CONFIG level on both Chassis devices and Compact devices. On Chassis devices, this command is available at the Interface level as well. This command is supported on Ethernet and Layer 3 interfaces.

1. Enter the global configuration to get access to interface mode.

```
device# configure terminal
```

2. Specify the interface to be configured in the interface mode.

```
device(config)# interface ethernet 1/3/11
```

3. Specify the threshold values for TCP SYN packets received per second.

```
device(config-if-e1000-1/3/11)# ip tcp burst-normal 10 burst-max 100 lockup 300
```

The following example configures the TCP/SYN attack protection at the virtual interface level or ve 1. For Layer 3 router code, if the interface is part of a VLAN that has a router VE, you must configure TCP/SYN attack protection at the VE level. When TCP/SYN attack protection is configured at the VE level, it will apply to routed traffic only. It will not affect switched traffic.

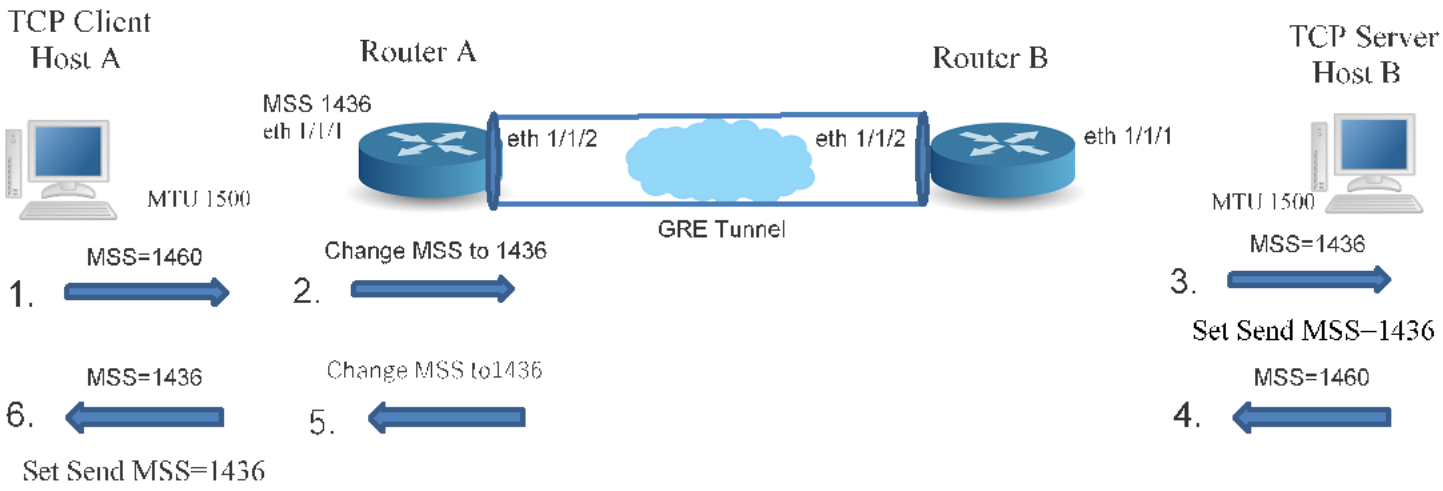
NOTE
You must configure VLAN information for the port before configuring TCP/SYN attack protection. You cannot change the VLAN configuration for a port on which TCP/SYN attack protection is enabled.

```
device# configure terminal
device(config)# vlan 2
device(config-vlan-2)# router-interface ve 1
device(config-vlan-2)# interface ve 1
device(config-vif-1)# ip tcp burst-normal 5000 burst-max 10000 lockup 300
```

TCP MSS Adjustment Overview

TCP MSS adjustment modifies the maximum segment size (MSS) field from the TCP header in the TCP SYN/SYN-ACK packet during a three-way handshake. TCP MSS adjustment applies only to TCP connections that pass through the router. It does not apply to TCP connections originating from or destined to the router.

Example



Host A is a TCP client and Host B is the TCP server, and there are two routers, A and B. Router A has TCP MSS adjustment configured on ingress port Ethernet 1/1/1 with a specific value because there is a GRE tunnel in the path, which will add an

additional 24 bytes of encapsulation. Router A modifies the TCP MSS adjustment value in the TCP SYN packet and forwards the packet. Host B receives the TCP SYN packet with the TCP MSS adjustment value set to the same value. Host B compares its MSS value to the received MSS value and sends the lower value to Host A as the "Send MSS" value. Host B send its own MSS value in the TCP SYN-ACK. In Router A, egress port Ethernet 1/1/1 is configured with the TCP MSS adjustment value. Router A modifies the MSS value in the TCP SYN-ACK and forwards the packet. Host A receives the TCP SYN-ACK and sends the "Send MSS" value to Host B.

Because Host A and Host B set a specific "Send MSS" value, neither of them will send a TCP data segment greater than a specific number of bytes to the other host. IP fragmentation of the TCP data packet is unnecessary though GRE tunnel encapsulation occurs at Router A and Router B.

The same outcome can be achieved by configuring TCP MSS adjustment in the GRE tunnel interface in Router A. In this case, the MSS will be modified only for the packets traversing the tunnel. The remaining traffic will not be modified.

Impact on Existing Functionality

The TCP SYN attack feature is impacted because of TCP MSS adjustment. Both features look for same fields in the packet, so either one of the ACL rules can be hit.

Because TCP MSS adjustment has inbuilt rate limiting applied, it can be viewed as a superset. Higher priority is given to TCP MSS adjustment when both features are configured on a Layer 3 interface.

TCP MSS Adjustment Limitations

- When TCP MSS adjustment is enabled on an interface, some delay in the TCP connection is established. Since we support MSS adjustment on ingress and egress traffic of an interface as well as for both SYN and SYN-ACK packets, 2 packets are trapped to CPU (Ingress SYN and egress SYN-ACK) for one TCP connection
- TCP MSS adjustment is not supported for OpenFlow packets.
- When TCP SYN attack and TCP MSS adjustment are configured on an interface, TCP MSS adjustment takes the higher priority.
- When PBR and TCP MSS adjustment both are applied, the TCP connections that go through PBR may be delayed, even if the outgoing interface of the PBR route does not have the TCP MSS adjustment configuration.
- When MSS is configured on the outgoing interface that PBR points to, but not on the outgoing interface that Layer 3 lookup points to, then the packet is not be trapped to the CPU. The MSS is not modified for the packet. For example, assume the normal route lookup for incoming packet with destination address 10.10.10.10 goes through 1/1/10. But PBR is configured for redirecting the packet with destination address 10.10.10.10 to 1/1/1. TCP MSS is configured on interface 1/1/1. Now MSS is not modified for the TCP SYN packet coming with destination address 10.10.10.10 even though it goes via MSS configured interface 1/1/1.
- TCP MSS adjustment ACL rules are given higher priority than the user-defined ACL rules. Therefore even if the user-defined rule is applied to drop the TCP packets, it comes to the CPU because of the TCP MSS adjustment rule and it is dropped in software.
- In case of ECMP routes for a destination, there is no guarantee that the packet will go through the path chosen in hardware. The outgoing TCP SYN/SYN-ACK packet will be trapped to the CPU and the MSS is modified according to the software route lookup.
- When TCP MSS adjustment is applied on a physical, VE, or LAG interface, it will modify the MSS only for the plain IP of IPv6 traffic.. It will not modify the MSS for tunneled or encrypted packets. The MSS value cannot be modified for an AH-only IPsec packet (unencrypted). The MD5/SHA-1 hash cannot be recomputed because the security keys to compute the hash value are unknown.

- TCP MSS adjustment configuration is not supported on loopback and management interfaces.
- TCP MSS adjustment is supported only on the Layer 3 interface.
- When TCP MSS is applied on physical, VE, or LAG interfaces, the MSS will be modified for plain IP and IPv6 traffic only. The MSS will not be modified for tunneled or encrypted packets.

Displaying statistics from a DoS attack

To display information about the dropped ICMP and TCP SYN packets due to increased burst thresholds, enter the **show statistics dos-attack** command.

```
device# show statistics dos-attack

---------------------------- Local Attack Statistics --------------------------
ICMP Drop Count    ICMP Block Count    SYN Drop Count    SYN Block Count
---------------    ----------------    --------------    ---------------
              0                   0                 0                   0
--------------------------- Transit Attack Statistics -------------------------
Port   ICMP Drop Count    ICMP Block Count    SYN Drop Count    SYN Block Count
-----  ---------------    ----------------    --------------    ---------------
1/3/11                 0                   0                 0                   0
```

Clear DoS attack statistics

To clear statistics about ICMP and TCP SYN packets, enter the **clear statistics dos-attack** command.

```
device# clear statistics dos-attack
```

IPv6 RA Guard

Securing IPv6 address configuration

In a IPv6 domain, a node can obtain an IPv6 address using the following two mechanisms:

- IPv6 address auto-configuration using router advertisements
- DHCPv6 protocol

In a typical man-in-middle (MiM) attack, the attacker can spoof as a router with spurious router advertisements.To prevent such attacks, IPv6 RA guard helps to secure the IPv6 address configuration in the network.

IPv6 RA guard overview

In an IPv6 network, devices are configured to send IPv6 Router Advertisements (RAs). Router advertisement and solicitation messages enable a node on a link to discover the routers on the same link. This helps the nodes to autoconfigure themselves on the network. Unintended misconfigurations or malicious attacks on the network lead to false RAs being present, which in turn causes operational problems for hosts on the network.

IPv6 RA guard improves security of the local IPv6 networks. The IPv6 RA guard is useful in network segments that are designed around a single Layer 2 switching device or a set of Layer 2 switching devices. You can configure IPv6 RA guard if you have local IPv6 networks and you are using auto-configuration for local addresses. IPv6 RA guard filters untrusted sources; host ports are dropped, and trusted ports are passed. The IPv6 RA guard filters RAs based on certain criteria.

You can configure RA guard policy and associate criteria such as whitelist, prefix list, and preference maximum value against which the RAs are inspected and the decision is taken whether to forward or drop the RA packets. You can configure a port as host, trusted, or untrusted. For the RA guard policy to take effect, you must configure the RA guard policy, and associate the criteria, and set the port type as host, trusted, or untrusted.

RA guard policy

An RA guard policy is a set of criteria against which the RAs are inspected by ports. Based on the RA guard policy configurations, RAs are forwarded or dropped. The whitelist, prefix-list, and maximum preference value configurations are set for a particular RA guard policy so that the RAs are inspected against all the criteria before being forwarded or dropped.

Before configuring an RA guard policy, you must enable ACL filtering based on VLAN membership using the **enable acl-per-port-per-vlan** command.

Whitelist

The whitelist contains the link-local addresses of the trusted sources; RAs from these sources can be forwarded. The RAs from the sources permitted by the whitelist are forwarded and the remaining RAs are dropped.

Prefix list

Prefix list is supported only on Layer 3 devices. The prefix list is configured at the global level using the **ipv6 prefix-list** command. IPv6 prefix lists can be used in the RA policy to inspect and restrict the advertised prefixes in the RA packets. RA packets from the trusted sources in the whitelist can be further inspected using the prefix list. If the RA packet has a prefix that does not match with the configured prefix list, the RA packet is dropped.

Maximum preference

RA packets may contain a router preference value. If the RA packets have a preference value higher the policy's maximum-preference value, the packets are dropped. If, for example, this value is set to medium and the advertised default router preference is set to high in the received packet, then the packet is dropped. If the option is set to medium or low in the received packet, then the packet is not dropped.

Trusted, untrusted, and host ports

IPv6 RA guard classifies interfaces on devices as trusted, untrusted, or host ports. For the configuration to take effect (trusted, untrusted, or host ports), the RA guard policy must be applied to the VLAN the ports are a part of. By default, all interfaces are configured as host ports. On a host port, all the RAs are dropped with a policy configured on the VLAN. Trusted ports are those that receive RAs within the network. Trusted ports allow received RAs to pass through without checking.

Depending on the configured policy settings, an RA packet is either forwarded through the interface or dropped. If you do not configure an RA guard policy on an untrusted or host port, all RAs are forwarded.

Configuration notes and feature limitations for IPv6 RA guard

- MAC filters and MAC-based VLANs are not supported with IPv6 RA guard.
- If an IPv6 ACL matching an ICMPv6 type RA packet is configured on an interface that is part of an RA guard-enabled VLAN, RA guard policy configuration takes precedence.
- IPv6 RA guard does not offer protection in environments where IPv6 traffic is tunneled.
- IPV6 RA guard can be configured on a switch port interface in the ingress direction and is supported only in the ingress direction; it is not supported in the egress direction.

Configuring IPv6 RA guard

- (Optional) Configure the IPv6 prefix list using the **ipv6 prefix-list** command (for a Layer 3 device) to associate a prefix list to an RA guard policy.
- Configure the **enable acl-per-port-per-vlan** command before you define an RA guard policy.

Configuring IPv6 RA guard includes the following steps:

1. Define an RA guard whitelist using the **ipv6 raguard whitelist** command. Add IPv6 addresses of all the sources from which the RA packets can be forwarded. You can create a maximum of 64 whitelists and each whitelist can have a maximum of 128 IPv6 address entries.
2. Define an RA guard policy using the **ipv6 raguard policy** command. You can configure a maximum of 256 RA guard policies.
3. Configure ports as trusted, untrusted, or host ports using the **raguard** command in the interface configuration mode.
4. Associate a whitelist with an RA guard policy using the **whitelist** command in the RA guard policy configuration mode. You can associate only one whitelist with an RA guard policy. If you do not associate a whitelist with an RA guard policy, all RA packets are dropped.
5. (Optional) (Only for Layer 3 devices) Associate an already defined prefix list with the RA guard policy using the **prefix-list** command in the RA guard policy configuration mode. You must provide the name of an IPv6 prefix list already configured using the **ipv6 prefix-list** command. Associate a prefix-list with an RA guard policy using the **prefix-list** command.
6. (Optional) Set the preference for RA packets using the **preference-maximum** command in the RA guard policy configuration mode.
7. Apply the RA guard policy to a VLAN using the **ipv6 raguard vlan** command in the global configuration mode. You can associate only one RA guard policy with a VLAN.
8. (Optional) Enable logging using the **logging** command in the RA guard policy configuration mode. If logging is enabled, you can verify the logs like RAs dropped, permitted, count for dropped packets, and reasons for the drop. Logging increases the CPU load and, for higher traffic rates, RA packets drop due to congestion if they are received at the line rate.
9. (Optional) Verify the RA guard configuration using the **show ipv6 raguard** command.
10. (Optional) Clear the RA packet counter using the **clear ipv6 raguard** command.
11. (Optional) Verify the RA packet counts using the **show ipv6 raguard counts command. Logging has to be enabled to verify the counts.**

Example of configuring IPv6 RA guard

The following sections describe how to configure IPv6 RA guard on a device or in a network.

Example: Configuring IPv6 RA guard on a device

The following example shows how to configure RA guard on a device.

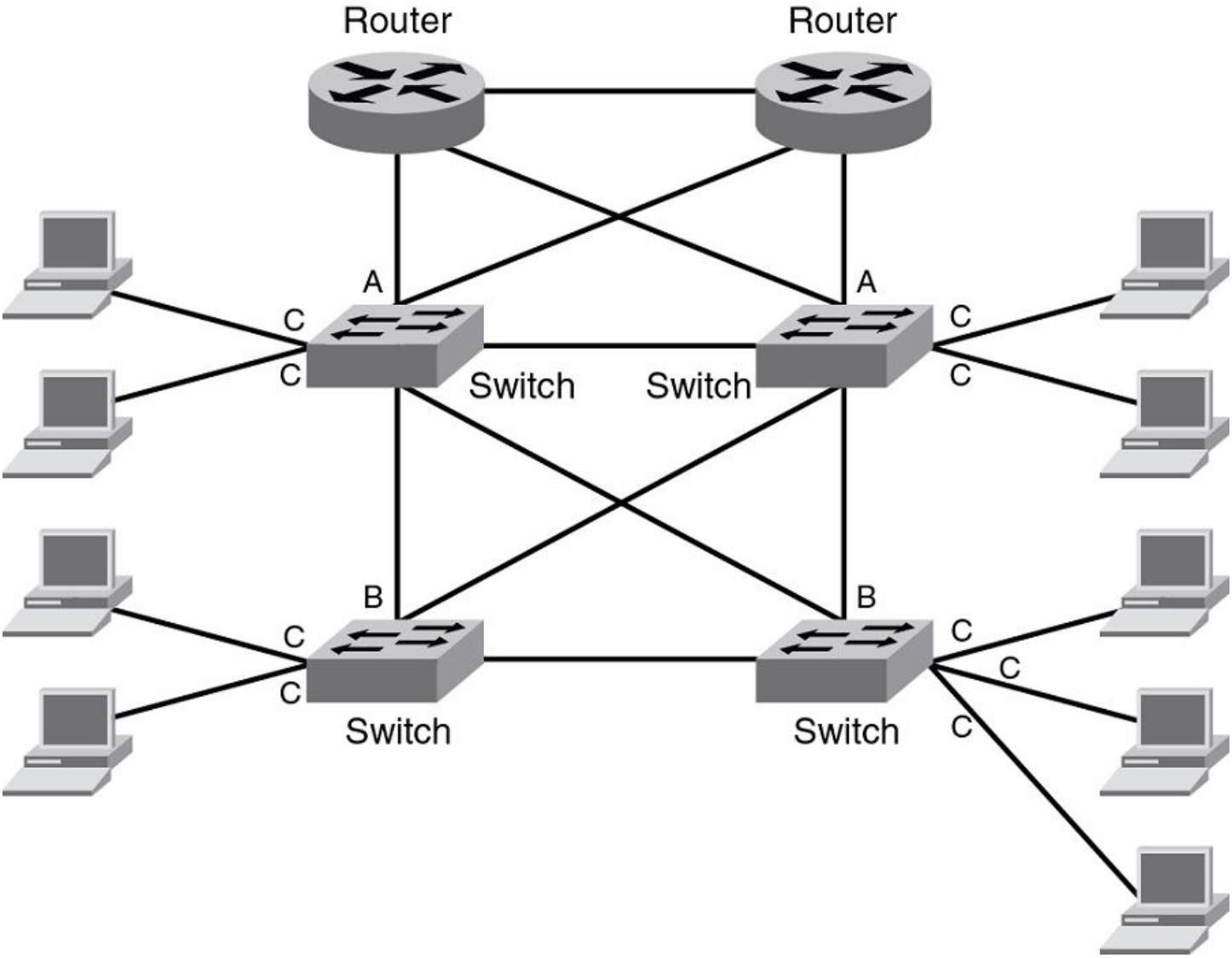
```
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:1
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:3
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:10
device(config)# ipv6 raguard policy policy1
device(ipv6-RAG-policy policy1)# whitelist 1
device(ipv6-RAG-policy policy1)# prefix-list raguard-prefix1
device(ipv6-RAG-policy policy1)# preference-maximum medium
device(ipv6-RAG-policy policy1)# logging
device(ipv6-RAG-policy policy1)# exit
device(config)# interface ethernet 1/1/1
device(config-int-e1000-1/1/1)# raguard untrusted
device(config-int-e1000-1/1/1)# exit
device(config)# ipv6 raguard vlan 1 policy policy1
```

```
device(config)# show ipv6 raguard all
device(config)# show ipv6 raguard counts all
```

Example: Configuring IPv6 RA guard in a network

The following example shows how to configure IPv6 RA guard on devices in a network. In this network topology, port A (ethernet 1/1/1) is configured as trusted, port B (ethernet 1/1/2) is configured as untrusted, and port C (ethernet 1/1/3) is configured as host. A whitelist is configured on port B.

FIGURE 35 IPv6 RA guard configuration in a network



Configuring port A:

Configure port A as a trusted port.

```
device(config)# interface ethernet 1/1/1
device(config-int-e1000-1/1/1)# raguard trust
```

Configuring port C:

On port C, create an RA Guard policy with no other options and associate the policy with a VLAN of which C is a member of. This helps block all RAs from C ports.

```
device(config)# ipv6 raguard policy policyC
device(ipv6-RAG-policy policyC)# exit
device(config)# ipv6 raguard vlan 1 policyC
```

Configuring port B:

On port B create an RA Guard policy with supported whitelist. This helps to permit RAs from only those sources. Associate a whitelist or prefix list with the RA guard policy.

```
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:10
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:5
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:12
device(config)# prefix-list raguard-prefix-list1 permit 2001:db8::/16
device(config)# ipv6 raguard policy policyB
device(ipv6-RAG-policy policyB)# whitelist 1
device(ipv6-RAG-policy policyB)# prefix-list raguard-prefix-list1
device(ipv6-RAG-policy policyB)# exit
device(config)# interface ethernet 1/1/2
device(config-int-e1000-1/1/2)# raguard untrust
device(config-int-e1000-1/1/2)# exit
device(config)# ipv6 raguard vlan 2 policyB
```

Example: Verifying the RA guard configuration

To view the RA guard packet counts, use the **show ipv6 raguard counts** command.

```
device# show ipv6 raguard counts policyB
DROPPED-host port:0
DROPPED-whitelist:3
DROPPED-prefixlist:1
DROPPED-max pref:1
DROPPED-trusted port:2
DROPPED-untrusted port:1
```

To verify the RA guard configuration, use the **show ipv6 raguard** command.

```
device# show ipv6 raguard all
policy:policyC
        whitelist:0
        max_pref:medium
policy:policyB
        whitelist:1
```

Joint Interoperability Test Command

JITC overview

The Joint Interoperability Test Command (JITC) mode on a FastIron device is compliant with the standards established by JITC, a United States military organization that tests technology pertaining to multiple branches of the armed services and the government.

The JITC mode implemented on a FastIron device enforces default behavior for some features to ensure strict JITC certification compliance.

AES-CTR encryption mode support for SSH

The Advanced Encryption Standard - Cipher Block Chaining (AES-CBC) encryption mode for Secure Shell (SSH) is vulnerable to certain plain-text attacks. The JITC mode uses AES-CTR (Counter) encryption mode for SSH instead of AES-CBC mode for enhanced security.

In the JITC mode, by default, the AES-CBC encryption mode for SSH is disabled and the AES-CTR (Counter) encryption mode is enabled. The **ip ssh encryption disable-aes-cbc** command that disables the AES-CBC mode can be seen in the running configuration. The encryption algorithms such as aes256-ctr, aes192-ctr, or aes128-ctr are enabled and the CBC mode ciphers are removed.

The AES-CBC mode can be re-enabled by issuing the **no ip ssh encryption disable-aes-cbc** command, which will bring back the pre-existing CBC ciphers (aes256-cbc, aes192-cbc, aes128-cbc, and 3des-cbc) along with the CTR ciphers.

NOTE
The AES-CTR mode must be configured both on the client and server sides to establish an SSH connection.

SHA1 authentication support for NTP

In the JITC mode, the symmetric key scheme supported for cryptographic authentication of messages uses the SHA1 keyed hash algorithm instead of the MD5 authentication scheme. The MD5 authentication for Network Time Protocol (NTP) is disabled by default in the JITC mode and the **disable authentication md5** command can be seen in the running configuration. Only the SHA1 authentication scheme is available to define the authentication key for NTP in the JITC mode. SHA1 authentication must be enabled manually using the **authentication-key key-id** command. In the JITC mode, only the SHA1 option is available.

The MD5 authentication scheme can be re-enabled by issuing the **no disable authentication md5** command. By doing so, the default JITC mode behavior is overridden.

IPv6 ACL for SNMPv3 group

As part of the JITC requirement, from 08.0.20a release onwards, the IPv6 access list is supported for the SNMPv3 group, and the incoming SNMP packets can be filtered based on the IPv6 ACL attached to the group.

For more information, refer to the "Defining an SNMP group" and "Defining an SNMP group and specifying which view is notified of traps" sections in *SNMP* chapter of the *Ruckus FastIron Management Configuration Guide*.

OpenSSL License

OpenSSL license

Copyright (c) 1998-2001 The OpenSSL Project. All rights reserved.

1. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
2. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
3. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation or other materials provided with the distribution.
4. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
5. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org .
6. Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.
7. Redistributions of any form whatsoever must retain the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)"

   THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT "AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Original SSLeay License

Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com) All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com). The implementation was written so as to conform with Netscapes SSL. This library is free for commercial and non-commercial use as long as the following conditions are aheared to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com). Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library

used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

1. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
2. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
3. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes cryptographic software written by Eric Young(eay@cryptsoft.com)" The word 'cryptographic' can be left out if the rouines from the library being used are not cryptographic related.
4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgment: "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

   THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. The licence and distribution terms for any publically available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence.

Keychain module

Keychain is a utility module that can be used by any application or routing protocol that uses authentication keys to establish secure communication with peers and validate the control packets.

Keychain module provides a secure infrastructure to prevent unauthorized routing updates to the network and ensures that only trusted routers participate in routing updates. Apart from authentication, keychain module provides a mechanism to ensure key rollover based on the lifetime or duration specified for each key used for authentication. The keys are rolled over to the next key if active and within the range of key IDs used by the protocols.

How keychain module works

Keychain module is independent of the protocols or applications that use it. The protocol packets use one of the active keys from the keychain profile. The keychain profile may have multiple keys with different attributes, such as authentication algorithms, passwords, and lifetimes.

NOTE
In FastIron 08.0.70, only OSPFv2 and OSPFv3 use the keychain module.

Each key in the keychain has a lifetime associated with it and a key is considered active if it is within the configured time range and has an authentication algorithm and a password. When a key expires, the keychain module notifies the application about the expiry of the key. This helps in changing the key periodically. Optionally, a tolerance value can be configured for the accept-keys and send-keys for the keychain to facilitate extension of the lifetime of the keys outside the active lifetime duration (prior to the start of the lifetime or after the end of the lifetime). The tolerance period helps to extend the period of the keys and works to smoothen the transition to a new key by the protocol. Keys cannot be used for authentication during the period that they are not active. Therefore, the lifetime of the keys must be configured so that the key activation periods of the keys overlap with each other to ensure that active keys are available at any time, which is critical for key rollover.

NOTE
All participating routers must have Network Time Protocol (NTP) enabled before setting the lifetime on the keys.

An application uses the keychain module to generate and send a message digest using the key and the specified algorithm and also validates that message digest it is carrying is correct while receiving the packets. When an application requests keys from the keychain module for sending and accepting the packets, the keychain module supplies all the active keys from the keychain and the application picks the desired key based on the inherent criteria of the protocol or application. The sending peer picks the key based on the lifetime and cryptographic algorithm, giving the application an option to choose the cryptographic algorithm that matches its criteria. The receiving peer decides on the key with which it authenticates based on the incoming key ID. When a keychain is configured under a protocol, all the packets generated by the protocol, such as routing updates and hello packets, are validated with that key ID. The same procedure is followed for receiving packets because the key ID will be used to validate the packets. It is imperative that the neighbors and participating routers have the same configuration at the other end.

Components of a keychain

Keychain module is a standalone infrastructure that can be used by any application that uses keys for authentication purposes to establish secure communication with peers.

A keychain is composed of the following components:

- Keychain profile: Each keychain is identified with a profile name. A maximum of up to 64 keychains can be configured.
- Key (key identifier): Keys can be added to the keychain profile by specifying key IDs. A maximum of 1024 keys can be configured across all the keychains. Each key ID within a keychain has its own properties, such as a password, authentication algorithm, send lifetime, and accept lifetime. A key is considered valid only if the key lifetime has not expired, and the password and authentication algorithm are specified. The range of returned key IDs usable varies with protocol. For each protocol, the key ID must be within a valid range. For example, the valid range of key IDs for OSPFv2 is 1 through 255. The application that uses the keychain module can reject the key IDs that are outside the permitted range. However, the keychain module does not place any restrictions in terms of user configuration of the key ID.
- Authentication algorithm: An authentication algorithm must be used for the key. The application or protocol chooses the cryptographic algorithm that matches its criteria. The following algorithms are the supported authentication algorithms:
  - HMAC-SHA-1
  - HMAC-SHA-256
  - MD5
  - SHA-1
  - SHA-256
- Password: Each key must have a password in encrypted form for the cryptographic algorithm.
- Lifetime of key: Each key has a lifetime for send and accept duration. Each key in the keychain has a lifetime associated with it and a key is considered active if it is within the configured time range. The lifetime of the key also depends on the tolerance value.
  - Accept lifetime: The time period during which the key on a keychain becomes active and is received as valid.
  - Send lifetime: The time period during which the key on a keychain becomes active and is valid to be sent.
- Tolerance: The tolerance value facilitates extension of the lifetime of the keys outside the active lifetime duration (prior to the start of the lifetime or after the end of the lifetime). If the tolerance value is configured, the start time of the key to become active is advanced (start time minus tolerance) and the end time is moved further ahead (end time plus tolerance) before the key expires, unless the end time is set to be infinite. A key is considered valid even when it is in the tolerance period.

OSPF keychain authentication

Applications such as OSPF can make use of keychain module that provides hitless authentication key rollover. Using the keychain module allows OSPF to overcome the limitation of a static configuration in authentication methods that requires manual intervention to change the key periodically.

For each OSPF protocol packet, a key is used to generate and verify a message digest. The key is valid for the entire duration of the protocol without any option to change the key string or authentication algorithm automatically. The keychain module that functions as a container of keys with different attributes such as an authentication algorithm, a password, and different lifetimes provides OSPF with an option to choose the key that best suits its criteria and automatically change the key ID, password, and

cryptographic algorithm without manual intervention. OSPFv2 and OSPFv3 authentication using the keychain can be configured using the **ip ospf authentication key-chain** and **ipv6 ospf authentication keychain** commands respectively. For more information on OSPFv2 and OSPFv3 authentication using keychain module, refer to the *Ruckus FastIron Layer 3 Routing Configuration Guide*.

Configuring a keychain module

Complete the following steps to configure keychain module.

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

   ```
   device# configure terminal
   ```

2. Configure a keychain profile.

   ```
   device(config)# keychain ruckus
   ```

3. (Optional) Configure the tolerance value for the keys.

   ```
   device(config-ruckus)# tolerance 10000
   ```

4. Configure a key by specifying a key identifier.

   ```
   device(config-ruckus)# key-id 1
   ```

5. Configure the authentication algorithm to be used for the key.

   ```
   device(config-ruckus-key-1)# authentication-algorithm md5
   ```

   Authentication algorithms such as HMAC-SHA-1, HMAC-SHA-256, MD5, SHA-1, and SHA-256 are supported. The application or protocol chooses the cryptographic algorithm that matches its criteria.

6. Configure the password to be used for the key.

   ```
   device(config-ruckus-key-1)# password abc
   ```

7. Configure the time period during which the key on a keychain becomes active and can be received as a valid key.

   ```
   device(config-ruckus-key-1)# accept-lifetime start 10-10-17 10:10:10 end 10000
   ```

   The expiry of the accept key can be configured as one of the following options: duration in seconds, infinite, or date and time format.

8. Configure the time period during which the key on a keychain becomes active and is valid to be sent.

   ```
   device(config-ruckus-key-1)# send-lifetime start 10-10-17 10:10:10 end infinite
   ```

   The expiry of the send key can be configured as one of the following options: duration in seconds, infinite, or date and time format.

Ruckus Wireless, Inc., a wholly owned subsidiary of CommScope, Inc.
350 West Java Dr., Sunnyvale, CA 94089 USA
www.ruckuswireless.com