CONFIGURATION GUIDE

Ruckus FastIron Security Configuration Guide, 08.0.40a

Supporting FastIron Software Release 08.0.40a

Part Number: 53-1003907-08
Publication Date: 27 April 2018

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

Contents

Preface

Document Conventions

The following tables list the text and notice 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.

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: docs@ruckuswireless.com

When contacting us, please include the following information:

- Document title and release number
- Document part number (on the cover page)
- Page number (if appropriate)
- For example:
  - Ruckus Small Cell Alarms Guide SC Release 1.3
  - Part number: 800-71306-001
  - Page 88

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 documentation by product or perform a text search. Access to Release Notes requires an active support contract and Ruckus Support Portal user account. Other technical documentation content is available without logging into 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 Networks products, and customers and partners with active support contracts.

For product support information and details on contacting the Support Team, go directly to the 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. Request 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, and Asia Pacific, toll-free numbers are available at https://support.ruckuswireless.com/contact-us and Live Chat is also available.

Self-Service Resources

The Support Portal at https://support.ruckuswireless.com/contact-us 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/software
- 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 includes descriptions of new information added to this guide for the FastIron 8.0.40a release.

TABLE 2 Summary of enhancements in FastIron release 8.0.40a

Feature	Description	Described in
IP Source guard	IP Source guard support is extended to 802.1X authentication-enabled port.	Support for IP source guard protection on page 221
Periodic reauthentication	Periodic reauthentication support is extended to MAC authentication-enabled ports.	Periodic reauthentication for 802.1X and MAC authenticated clients on page 213
DHCP snooping	DHCP snooping is interoperable with Flexible authentication.	DHCP snooping support for Flexible authentication on page 214
Dynamic ARP inspection	Dynamic ARP Inspection (DAI) is interoperable with Flexible authentication.	Dynamic ARP Inspection support for Flexible authentication on page 213
PBR support on Brocade ICX 7250	Policy-Based Routing is supported on Brocade ICX 7250.	Policy-Based Routing on page 165

Supported hardware

This guide supports the following product families from Ruckus:

- ICX 7750 Series
- ICX 7450 Series
- ICX 7250 Series

For information about the specific models and modules supported in a product family, refer to the hardware installation guide for that product family.

How command information is presented in this 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 20.

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 16 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 23.

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. You can assign a password for Telnet access using one of the following methods.

Set the password "letmein" for Telnet access to the CLI using the following command at the global CONFIG level.

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

Syntax: [no] enable telnet password *string*

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 can optionally 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, use the following CLI method.

To suppress the connection rejection message sent by the device to a denied Telnet client, enter the following command at the global CONFIG level of the CLI.

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

Syntax: [no] 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 also can 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 23.

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 will validate 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 83.

Follow the steps given below to set passwords for management privilege levels.

1. At the opening CLI prompt, enter the following command to change to the Privileged level of the EXEC mode.

```
device> enable
device#
```

2. Access the CONFIG level of the CLI by entering the following command.

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

3. Enter the following command to set the Super User level password.

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

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 cannot begin with a number.

4. Enter the following commands to set the Port Configuration level and Read Only level passwords.

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

Syntax: enable super-user-password *text*

Syntax: enable port-config-password *text*

Syntax: enable read-only-password *text*

NOTE
If you forget your Super User level password, refer to Recovering from a lost password on page 22.

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 User EXEC and Privileged EXEC levels
  - The port-specific parts of the CONFIG level
  - All interface configuration levels
- Read Only level gives access to:
  - The User EXEC and Privileged EXEC levels

You can grant additional access to a privilege level on an individual command basis. To grant the additional access, you 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 on the CLI.

Enhance the Port Configuration privilege level so users also can enter IP commands at the global CONFIG level.

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

In this command, **configure** specifies that the enhanced access is for a command at the global CONFIG level of the CLI. The **level 4** parameter indicates that the enhanced access is for management privilege level 4 (Port Configuration). All users with Port Configuration privileges will have the enhanced access. 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 CONFIG level.

Syntax: [no] privilege *cli-level* **level** *privilege-level command-string*

The cli-level parameter specifies the CLI level and can be one of the following values:

- **exec** - EXEC level; for example, device> or device#
- **configure** - CONFIG level; for example, device(config)#
- **interface** - Interface level; for example, device(config-if-6)#
- **loopback-interface** - loopback interface level
- **virtual-interface** - Virtual-interface level; for example, device(config-vif-6)#
- **dot1x** - 802.1X configuration level
- **ipv6-access-list** - IPv6 access list configuration level
- **rip-router** - RIP router level; for example, device(config-rip-router)#
- **ospf-router** - OSPF router level; for example, device(config-ospf-router)#
- **dvmrp-router** - DVMRP router level; for example, device(config-dvmrp-router)#
- **pim-router** - PIM router level; for example, device(config-pim-router)#
- **bgp-router** - BGP4 router level; for example, device(config-bgp-router)#
- **vrrp-router** - VRRP configuration level
- **gvrp** - GVRP configuration level
- **trunk** - trunk configuration level
- **port-vlan** - Port-based VLAN level; for example, device(config-vlan)#
- **protocol-vlan** - Protocol-based VLAN level

The privilege-level indicates the number of the management privilege level you are augmenting. You can specify one of the following:

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

The command *-string* parameter specifies the command you are allowing users with the specified privilege level to enter. To display a list of the commands at a CLI level, enter "?" at that level's command prompt.

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(config)#enable password-display
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 file and running-config. When the **enable password-display** command is configured, the user password and snmp community string are encrypted in the **show run** command output. Enter the command at the global CONFIG 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. You can configure the device to require that Line, Enable, and Local passwords be at least a specified length.

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

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

Syntax: enable password-min-length *number-of-characters*

The number-of-characters can be from 1 - 48.

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 20.

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 83.

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 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 23.

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 an enable and 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(config)#enable strict-password-enforcement
```

Syntax: [no] 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 will be 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 "Mail" were used consecutively in the previous password
- &3B9aYa& because "aYa&" were used consecutively in the previous password
- i4aYEv#8 because "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 will not be allowed and the following message will be 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 actual password characters entered.

The following shows the default CLI behavior when configuring a username and password.

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

The following shows the CLI behavior when configuring a username and password when **password-masking** is enabled.

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

NOTE
When password masking is enabled, press the [Enter] key before entering the password.

Syntax: username *name* **password** [**Enter**]

For [**Enter**], press the Enter key. Enter the password when prompted.

If **strict-password-enforcement** is enabled, enter a password which contains the required character combination. Refer to Enabling enhanced user password combination requirements on page 24.

To enable password masking, enter the following command.

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

Syntax: [no] 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 will automatically prompt 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 configuration** command, indicated by set-time.

```
device# show run
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, then set-time will appear as "set-time 0" in the output of the **show running configuration** command.

A username set-time configuration is removed when:

- 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 configuration** output.

Note that if a username does not have an assigned password, the username will not have a set-time configuration.

Password aging is disabled by default. To enable it, enter the following command at the global configuration level of the CLI.

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

Syntax: [no] 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 will prompt the user to choose a different password.

To configure enhanced password history, enter a command such as the following at the global configuration level of the CLI.

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

Syntax: [no] enable user password-history *previous-passwords*

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 Brocade 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 a command such as the following at the global configuration level of the CLI.

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

Syntax: username *name* **expires** *days*

The *name* variable is the username of the user. The *days* variable is 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
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 a command similar to the following at the global configuration level of the CLI.

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

Syntax: [no] username *user-string* **privilege** *privilege-level* **nopassword**

Local user accounts with unencrypted passwords

If you want to use unencrypted passwords for local user accounts, enter a command such as the following at the global configuration level of the CLI.

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

If password masking is enabled, press Enter before entering the password.

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

This command adds a local user account with the user name "wonka" and the password. This account has the Super User privilege level; this user has full access to all configuration and display features.

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

This command adds a user account for username "waldo", the password "whereis", and with the Read Only privilege level. This user can look for information but cannot make configuration changes.

Syntax: [no] username *user-string* **privilege** *privilege-level* [**password** *password-string* | **nopassword**]

You can enter up to 48 characters for *user-string* variable.

The **privilege** *privilege-level* 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**.

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
```

Syntax: show users

To know the different methods to secure access to the device using the configured username and password, see Authentication-method lists on page 83.

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.

Changing the SSL server certificate key size

The default key size for Brocade-issued and imported digital certificates is 1024 bits. If desired, you can change the default key size to a value of 512, 2048, or 4096 bits.

To do so, enter a command such as the following at the Global CONFIG level of the CLI.

```
Brocade(config)#ip ssl cert-key-size 512
```

Syntax: **ip ssl cert-key-size** *512/ 1024/ 2048/ 4096*

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

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 Brocade 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(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
```

Syntax: telnet access-group *num*

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

The commands above configure ACL 10, then apply the ACL as the access list for Telnet access. The device allows Telnet access to all IP addresses except those listed in ACL 10.

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

```
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 this 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
```

Syntax: ssh access-group *num*

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 **ssh access-group 10** 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.

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

Syntax: **web access-group** *num*

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

To restrict SNMP access to the device using ACLs, enter commands such as the following.

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

```
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
```

Syntax: snmp-server community *string* [**ro** | **rw**] *num*

The string parameter specifies the SNMP community string the user must enter to gain SNMP access.

The **ro** parameter indicates that the community string is for read-only ("get") access. The **rw** parameter indicates the community string is for read-write ("set") access.

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

These commands configure ACLs 25 and 30, then apply the ACLs to community strings.

ACL 25 is used to control read-only access using the "public" community string. ACL 30 is used to control read-write access using the "private" community string.

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

Defining the console idle time

By default, a Ruckus device does not time out serial console sessions. A serial session remains open indefinitely until you close it. You can however 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 following command.

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

Syntax: [no] console timeout [*0-240*]

Possible values: 0 - 240 minutes

Default value: 0 minutes (no timeout)

> **NOTE**
> In RADIUS, the standard attribute Idle-Timeout is used to define the console session timeout value. The attribute 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

To allow Telnet access to the Ruckus device only to the host with IP address 10.157.22.39, enter the following command.

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

Syntax: [no] telnet client { *ip-addr* | *ipv6-addr* }

Restricting SSH access to a specific IP address

To allow SSH access to the Ruckus device only to the host with IP address 10.157.22.39, enter the following command.

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

Syntax: [no] ip ssh client { *ip-addr* | *ipv6-addr* }

Restricting Web management access to a specific IP address

To allow Web management access to the Brocade device only to the host with IP address 209.157.22.26, enter the following command.

```
Brocade(config)#web-client 209.157.22.26
```

Syntax: **[no] web-client** { *ip-addr* | *ipv6-addr* }

Restricting SNMP access to a specific IP address

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
```

Syntax: [no] snmp-client { *ip-addr* | *ipv6-addr* }

Restricting all remote management access to a specific IP address

To allow Telnet and SNMP management access to the Ruckus device only to the host with IP address 10.157.22.69, enter three separate commands (one for each access type) or enter the following command.

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

Syntax: [no] all-client { *ip-addr* | *ipv6-addr* }

Restricting access to the device based on IP orMAC 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(config)#telnet client 10.157.22.39 0000.000f.e9a0
```

Syntax: [no] telnet client { *ip-addr* | *ipv6-addrmac-addr* }

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

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

Syntax: [no] telnet client any *mac-addr*

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 command.

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

Syntax: [no] ip ssh client { *ip-addr* | *ipv6-addrmac-addr* }

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

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

Syntax: [no] ip ssh client any *mac-addr*

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, use the following command.

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

Syntax: [no] telnet timeout *minutes*

For minutes *enter a value from 0 - 240. The default value is 0 minutes (no timeout).*

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, use the following command.

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

Syntax: [no] telnet login-timeout *minutes*

For minutes , *enter a value from 1 to 10. The default timeout period is 1 minute.*

Specifying the maximum number of login attemptsfor Telnet access

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.

You can specify the number of attempts a Telnet user has to enter a correct username and password before the device disconnects the Telnet session. For example, to allow a Telnet user up to 5 chances to enter a correct username and password, enter the following command.

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

Syntax: [no] telnet login-retries *number*

You can specify from 0 - 5 attempts. The default is 4 attempts.

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

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 the methods listed above 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 who 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 a command such as the following.

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

The command in this 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.

Syntax: [no] telnet server enable vlan *vlan-id*

Restricting Web management access to a specific VLAN

To allow Web management access only to clients in a specific VLAN, enter a command such as the following.

```
Brocade(config)#web-management enable vlan 10
```

The command in this 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.

Syntax: **[no] web-management enable vlan** *vlan-id*

Restricting SNMP access to a specific VLAN

To allow SNMP access only to clients in a specific VLAN, enter a command such as the following.

```
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.

Syntax: [no] snmp-server enable vlan *vlan-id*

Restricting TFTP access to a specific VLAN

To allow TFTP access only to clients in a specific VLAN, enter a command such as the following.

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

The command in this 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.

Syntax: [no] tftp client enable vlan *vlan-id*

Designated VLAN for management sessions to a Layer 2 switch

All Brocade FastIron 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 also can 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 the following.

```
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
```

These commands configure port-based VLAN 10 to consist of ports 1/1/1 - 1/1/4 and to be the designated management VLAN. The last two commands configure default gateways for the VLAN. Since the 10.10.10.1 gateway has a lower metric, the software uses this gateway. The other gateway remains in the configuration but is not used. You can use the other one by changing the metrics so that the 10.20.20.1 gateway has the lower metric.

Syntax: [no] default-gateway *ip-addr metric*

The *ip-addr* parameters specify the IP address of the gateway router.

The *metric* parameter specifies the metric (cost) of the gateway. You can specify a value from 1 - 5. There is no default. The software uses the gateway with the lowest metric.

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.

Allowing SSHv2 access to the Brocade device

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

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

Syntax: crypto key [**generate** | **zeroize**]

The **generate** parameter generates a dsa key pair.

The **zeroize** parameter deletes the currently operative dsa key pair.

In addition, you must use AAA authentication to create a password to allow SSHv2 access. For example the following command configures AAA authentication to use TACACS+ for authentication as the default or local if TACACS+ is not available.

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

Allowing SNMP access to the Brocade device

To allow SNMP access to the Ruckus device, enter the following command.

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

Syntax: [no] snmp server

Allowing Web management through HTTP for the Brocade device

To allow web management through HTTP for the Brocade device, you enable web management as shown in the following command.

```
Brocade(config)#web-management http
```

Syntax: **[no] web-management** [**http** | **https**]

When using the web-management command, specify the **http** or **https** parameters.

The **http** parameter specifies that web management is enabled for HTTP access.

The **https** parameter specifies that web management is enabled for HTTPS access.

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 Brocade device on page 38. Additionally, you must generate a crypto SSL certificate or import digital certificates issued by a third-party Certificate Authority (CA).

To generate a crypto SSL certificate use the following command.

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

Syntax: **crypto-ssl certificate** [**generate** | **zeroize**]

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

The **generate** parameter generates an ssl certificate.

The **zeroize** parameter deletes the currently operative ssl certificate.

To import a digital certificate issued by a third-party Certificate Authority (CA) and save it in the flash memory, use the following command.

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

Syntax: **ip ssl certificate-data-file tftp** *ip-addr file-name*

The *ip-addr* variable is the IP address of the TFTP server from which the digital certificate file is being downloaded.

The *file-name* variable is the file name of the digital certificate that you are importing to the router.

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 applications.

Disabling Telnet access

You can use a Telnet client to access the CLI on 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(config)#no telnet server
```

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

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

Syntax: [no] 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.

```
Brocade(config)#no web-management
```

Syntax: **[no] web-management** [**http** | **https**]

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 Brocade 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 with the command. Here is an example.

```
Brocade(config)#no web-management hp-top-tools
```

Syntax: **[no] web-management** [**allow-no-password** | **enable** [**vlan** *vlan-id*] | **front-panel** | **hp-top-tools** | **list-menu**]

The **hp-top-tools** parameter disables TCP port 280.

Disabling SNMP access

To disable SNMP management of the device.

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

To later re-enable SNMP management of the device.

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

Syntax: [no] snmp-server

Disabling TFTP access

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

To disable TFTP client access, enter the following command at the Global CONFIG level of the CLI.

```
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 users enter this command while TFTP is disabled, the system will reject the command and display an error message.

To re-enable TFTP client access once it is disabled, enter the following command.

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

Syntax: [no] tftp disable

TACACS and TACACS+ 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 51.

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 authorization commands 0 default  tacacs+
aaa authorization exec default  tacacs+
aaa accounting commands 0 default start-stop  tacacs+
aaa accounting exec default start-stop  tacacs+
aaa accounting system default start-stop  tacacs+
enable aaa console
hostname Fred
ip address 10.10.6.56/255
tacacs-server host 255.253.255
tacacs-server key 2 $d3NpZ0BVXFpJ
```

kill console

Syntax: kill console [**all** | **unit**]

- **all** - logs out all console port on stack units that are not the Active Controller
- **unit** - logs out the console port on a specified unit

Once AAA console is enabled, you should log out any open console ports on your traditional stack using the **kill console** command:

```
device(config)#kill console all
```

In case a user forgets to log out or a console is left unattended, you can also configure the console timeout (in minutes) for active stack units.

```
device(config)#stack unit 1
device(config-unit-1)#console timeout 5
```

NOTE
Console timeout will not work for non-active and member units, use “kill console” command to log out all console sessions for non-active and member units. The sessions will re-authenticate for console access.

Use the **show who** and the **show telnet** commands to confirm the status of console sessions.

```
stack9#show who
Console connections (by unit number):
 1      established
        you are connecting to this session
        4 seconds in idle
 2      established
        1 hours 3 minutes 12 seconds in idle
 3      established
        1 hours 3 minutes 9 seconds in idle
 4      established
        1 hours 3 minutes 3 seconds in idle
Telnet connections (inbound):
 1      closed
 2      closed
 3      closed
 4      closed
 5      closed
Telnet connection (outbound):
 6      closed
SSH connections:
 1      closed
 2      closed
 3      closed
 4      closed
 5      closed
stack9#
stack9#show telnet
Console connections (by unit number):
 1      established
        you are connecting to this session
        1 minutes 5 seconds in idle
 2      established
        1 hours 4 minutes 18 seconds in idle
 3      established
        1 hours 4 minutes 15 seconds in idle
 4      established
        1 hours 4 minutes 9 seconds in idle
Telnet connections (inbound):
 1      closed
 2      closed
 3      closed
 4      closed
 5      closed
Telnet connection (outbound):
 6      closed
SSH connections:
 1      closed
 2      closed
 3      closed
 4      closed
 5      closed
stack9#
```

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

User action	Applicable AAA operations
	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, not both.

Configuring TACACS

Follow the procedure given below for TACACS configurations.

1. Identify TACACS servers. Refer to Identifying the TACACS/TACACS+ servers on page 47.
2. Set optional parameters. Refer to Setting optional TACACS and TACACS+ parameters on page 48.
3. Configure authentication-method lists. Refer to Configuring authentication-method lists forTACACS and TACACS+ on page 50.

Configuring TACACS+

Follow the procedure given below for TACACS+ configurations.

1. Identify TACACS+ servers. Refer to Identifying the TACACS/TACACS+ servers on page 47.
2. Set optional parameters. Refer to Setting optional TACACS and TACACS+ parameters on page 48.
3. Configure authentication-method lists. Refer to Configuring authentication-method lists forTACACS and TACACS+ on page 50.
4. Optionally configure TACACS+ authorization. Refer to Configuring TACACS+ authorization on page 52.
5. Optionally configure TACACS+ accounting. Refer to TACACS+ accounting configuration on page 55.

Enabling TACACS

TACACS is disabled by default. To configure TACACS/TACACS+ authentication parameters, you must enable TACACS by entering the following command.

```
device(config)#enable snmp config-tacacs
```

Syntax: [no] enable snmp [**config-radius** | **config-tacacs**]

The config-radius parameter specifies the RADIUS configuration mode. RADIUS is disabled by default.

The config-tacacs parameter specifies the TACACS configuration mode. TACACS is disabled by default.

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
```

Syntax: tacacs-server host { *ip-addr* |*ipv6-addr* | *server-name* } [**auth-port** *number*] [**acct-port**number]

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.

1. 10.94.6.161
2. 10.94.6.191

3. 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 50.) 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

In a TACACS+ configuration, you can designate a server to handle a specific AAA task. 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
```

Syntax: tacacs-server host { *ip-addr* | *ipv6-addr* | *server-name* } [**auth-port** *num*] [**authentication-only** | **authorization-only** | **accounting-only** | **default**] [**key** [**0** | **1**] *string*]

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
```

Syntax: tacacs-server key [**0**] *string*

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
```

Syntax: tacacs-server retransmit *number*

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
```

Syntax: tacacs-server timeout *number*

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.

Syntax: [no] aaa authentication { **enable** | **login default** } *method 1* [*method 2-7*]

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 Brocade 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 3 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 20.
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 TACACS/TACACS+, refer to Authentication-method lists on page 83.

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
```

Syntax: 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
```

Syntax: [no] 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+
```

Syntax: aaa authorization exec default tacacs+[**none**]

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**
> 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 4 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+
```

Syntax: aaa authorization commands *privilege-level* **default** [**tacacs+** | **radius** | **none**]

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
```

Syntax: [no] 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+
```

Syntax: aaa accounting exec default start-stop [**tacacs+** | **radius** | **none**]

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.

Syntax: aaa accounting commands *privilege-level* **default start-stop** [**radius** | **tacacs+** | **none**]

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+
```

Syntax: aaa accounting system default start-stop [**radius** | **tacacs+** | **none**]

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 Switch.

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 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 5 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 • 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

```
Brocade#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
```

Syntax: **show web connection**

Use the following command to clear web connections:

```
Brocade#clear web-connection
```

Syntax: **clear web connection**

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

```
Brocade#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

RADIUS authentication, authorization, and accounting

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 a command he or she has entered, as well as accounting , which causes the Ruckus device to log information on a RADIUS accounting server when specified events occur on the device.

FastIron supports TLS encryption for RADIUS server and client authentication.

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 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 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 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 operations that take place when a user gains access to a Ruckus device that has RADIUS 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
	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 enters the command: [no] aaa accounting system defaultstart-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

User action	Applicable AAA operations
	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

- You must deploy at least one RADIUS server in your network.
- 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. In earlier releases, all configured RADIUS servers are "global" servers and apply to users on all ports of the device. Refer to RADIUS server per port on page 67.
- 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 it is mapped. If there are no RADIUS servers mapped to a port, it will use the "global" servers for authentication. In earlier releases, all RADIUS servers are "global" servers and cannot be bound to individual ports. Refer to RADIUS server to individual ports mapping on page 68.
- 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 backup authentication methods for each access type.

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 Ruckus-specific attributes on the RADIUS server on page 63.
2. Identify the RADIUS server to the Ruckus device. Refer to Identifying the RADIUS server to the Ruckus device on page 66.
3. Optionally specify different servers for individual AAA functions. Refer to Specifying different servers for individual AAA functions on page 66.
4. Optionally configure the RADIUS server as a "port only" server. Refer to RADIUS server per port on page 67.
5. Optionally bind the RADIUS servers to ports on the Ruckus device. Refer to RADIUS server to individual ports mapping on page 68.
6. Set RADIUS parameters. Refer to RADIUS parameters on page 68.
7. Configure authentication-method lists. Refer to Setting authentication-method lists for RADIUS on page 70.
8. Optionally configure RADIUS authorization. Refer to RADIUS authorization on page 71.
9. Optionally configure RADIUS accounting. Refer to RADIUS accounting on page 73.

Ruckus-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 Ruckus vendor-specific attributes that indicate:

- 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 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 Ruckus vendor-specific attributes.

TABLE 6 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** - S**uper 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

TABLE 6 Ruckus vendor-specific attributes for RADIUS (continued)

Attribute name	Attribute ID	Data type	Description
			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]

TABLE 6 Ruckus vendor-specific attributes for RADIUS (continued)

Attribute name	Attribute ID	Data type	Description
			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 **4** - qos_priority_4 **5** - qos_priority_5 **6** - qos_priority_6 **7** - qos_priority_7

Enabling SNMP to configure RADIUS

To enable SNMP access to RADIUS MIB objects on the device, enter a command such as the following.

```
device(config)#enable snmp config-radius
```

Syntax: [no] enable snmp [**config-radius** |**config-tacacs**]

The *config-radius* parameter specifies the RADIUS configuration mode. RADIUS is disabled by default.

The *config-tacacs* parameter specifies the TACACS configuration mode. TACACS is disabled by default.

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.

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

Syntax: radius-server host { *ip-addr* |*ipv6-addr* | *hostname* } [**auth-port** *number*]

The **host** *ip-addr* | *ipv6-addr* | *server-name* parameter is either an IP address or an ASCII text string.

The *auth-port* parameter is the Authentication port number. The default is 1812.

The *acct-port* parameter is the Accounting port number. The default is 1813.

Specifying different servers for individual AAA functions

In a RADIUS configuration, you can designate a server to handle a specific AAA task. 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, authorization, 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 10.2.3.5 authorization-only key def
device(config)# radius-server host 10.2.3.6 accounting-only key ghi
```

Syntax: radius-server host { *ip-addr* | *ipv6-addr* | *server-name* } [**ssl-auth-port** *number* | **auth-port** *number*] [**acct-port** *number*] [**authentication-only** | **authorization-only** | **accounting-only** | **default**] [**key** { [**0** | **2**] *string* }]

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

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 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
```

The above configuration has the following affect:

- RADIUS servers 10.10.10.103 and 10.10.10.104 will be 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 68.
- RADIUS servers 10.10.10.105 and 10.10.10.106 will be 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, will send a RADIUS request to the first configured RADIUS server, 10.10.10.105. If the request fails, it will go to the second configured RADIUS server, 10.10.10.106. It will not send requests to 10.10.10.103 or 10.10.10.104, since these servers are configured as port servers.

Syntax: radius-server host { *ip-addr* | *server-name* } [**auth-port** *number* | **ssl-auth-port** *number*] [**acct-port**number] [**default key** *string* **dot1x**] [**port-only**]

The **host** *ip-addr* is the IPv4 address.

The **auth-port** *number* parameter is the Authentication port number; it is an optional parameter. The default is 1812.

The **ssl-auth-port** *number* specifies that the server is a RADIUS server running over a TLS-encrypted TCP session. Only one of 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.

The **acct-port** *number* parameter is the Accounting port number; it is an optional parameter. The default is 1813.

The **default key** *string* **dot1x** parameter indicates that this RADIUS server supports the 802.1X standard. A RADIUS server that supports the 802.1X standard can also be used to authenticate non-802.1X authentication requests.

The **port-only** parameter is optional and specifies that the server will be used only to authenticate users on ports to which it is mapped.

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

To map a RADIUS server to a port, enter commands such as the following.

```
device(config)#int e 3
device(config-if-e1000-3)#dot1x port-control auto
device(config-if-e1000-3)#use-radius-server 10.10.10.103
device(config-if-e1000-3)#use-radius-server 10.10.10.110
```

With the above configuration, port e 3 would send a RADIUS request to 10.10.10.103 first, since it is the first server mapped to the port. If it fails, it will go to 10.10.10.110.

Syntax: use-radius-server *ip-addr*

The **host** *ip-addr* is an IPv4 address.

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

Setting the retransmission limit

The **retransmit** parameter specifies the maximum number of retransmission attempts. When an authentication request times out, the Ruckus software will retransmit the request up to the maximum number of retransmissions configured. The default retransmit value is 3 retries. The range of retransmit values is from 1 - 5.

To set the RADIUS retransmit limit, enter a command such as the following.

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

Syntax: tacacs-server retransmit *number*

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

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

Syntax: radius-server timeout *number*

Setting RADIUS over IPv6

Ruckus devices support the ability to send RADIUS packets over an IPv6 network.

The ipv6-host address is the IPv6 address of the RADIUS server. When you enter the IPv6 host address, you do not need to specify the prefix length. A prefix length of 128 is implied.

To enable the Ruckus device to send RADIUS packets over IPv6, enter a command such as the following at the Global CONFIG level of the CLI.

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

Syntax: radius-server host ipv6 *ipv6-host-address*

Setting authentication-method lists for RADIUS

You can use RADIUS to authenticate Telnet/SSH access and access to Privileged EXEC level 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 backup authentication methods are specified as alternates. If RADIUS 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 RADIUS, you must create a separate authentication-method list for Telnet or SSH CLI access and for CLI access to the Privileged EXEC level and CONFIG levels of the CLI.

To create an authentication-method list that specifies RADIUS as the primary authentication method for securing Telnet access to the CLI.

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

The commands above cause RADIUS to be the primary authentication method for securing Telnet access to the CLI. If RADIUS authentication fails due to an error with the server, local authentication is used instead.

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

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

The command above causes RADIUS to be the primary authentication method for securing access to Privileged EXEC level and CONFIG levels of the CLI. If RADIUS 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.

Syntax: [no] aaa authentication { **enable** | **login default** } *method 1* [*method 2-7*]

The **aaa authentication** | **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 Brocade 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 7 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 20.

TABLE 7 Authentication method values (continued)

Method parameter	Description
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 83.

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
```

Syntax: 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
```

Syntax: [no] 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
```

Syntax: aaa authorization exec default [**radius** | **none**]

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 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 radius
```

Syntax: aaa authorization commands *privilege-level* **default** [**tacacs+** | **radius** | **none**]

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

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

Syntax: [no] enable aaa 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.

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.

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

Syntax: aaa accounting exec default start-stop [**radius** | **tacacs+** | **none**]

Configuring RADIUS accounting for CLI commands

You can configure RADIUS accounting for CLI commands by specifying a privilege level whose commands require accounting. For example, 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(config)#aaa accounting commands 0 default start-stop radius
```

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.

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.

Syntax: aaa accounting commands *privilege-level* **default start-stop** [**radius** | **tacacs** | **none**]

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 RADIUS accounting for system events

You can configure RADIUS 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 RADIUS 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 radius
```

Syntax: aaa accounting system default start-stop [**radius** | **tacacs+** | **none**]

Configuring an interface as the source for all RADIUS 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 RADIUS packets from the Layer 3 Switch.

RADIUS dynamic authorizations

Adds two new packets to the current RADIUS standard.

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 or 802.1x 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 Brocade 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

Describes enabling RADIUS CoA and Disconnect Message handling.

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

1. Enter global configuration mode.
2. Enter the **aaa authorization coa enable**command.

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

3. Configure the key between the CoA client and the device with the **radius-client coa host** command

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

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 8 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 9 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 10 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

TABLE 10 Output of the show aaa command for RADIUS (continued)

Field	Description
	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

802.1X accounting configuration

802.1X accounting enables the recording of information about 802.1X clients who were successfully authenticated and allowed access to the network. When 802.1X accounting is enabled on the Ruckus device, it sends the following information to a RADIUS server whenever an authenticated 802.1X 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

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 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 802.1X 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.

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** CLI command)

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

802.1X Accounting attributes for RADIUS

Ruckus devices support the following RADIUS attributes for 802.1X accounting.

TABLE 11 802.1X accounting attributes for RADIUS

Attribute name	Attribute ID	Data Type	Description
Acct-Session-ID	44	Integer	The account session ID, which is a number from 1 to 4294967295.
Acct-Status-Type	40	integer	Indicates whether the accounting request marks the beginning (start) or end (stop) of the user service.

TABLE 11 802.1X accounting attributes for RADIUS (continued)

Attribute name	Attribute ID	Data Type	Description
			1 - Start 2 - Stop
Calling-Station-Id	31	string	The supplicant MAC address in ASCII format (upper case only), with octet values separated by a dash (-). For example 00-00-00-23-19-C0
NAS-Identifier	32	string	The hostname of the device. Here NAS stands for "network access server".
NAS-Port	5	integer	The physical port number. Here NAS stands for "network access server".
NAS-Port-Type	61	integer	The physical port type. Here NAS stands for "network access server".
user-name	1	string	The user name.

Enabling 802.1X accounting

To enable 802.1X accounting, enter the following command.

```
device(config)#aaa accounting dot1x default start-stop radius none
```

Syntax: aaa accounting dot1x default start-stop { **radius** | **none** }

radius - Use the list of all RADIUS servers that support 802.1X for authentication.

none - Use no authentication. The client is automatically authenticated without the device using information supplied by the client.

> **NOTE**
> If you specify both **radius** and **none** , make sure **radius** comes before **none** .

SSL security

The Brocade 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 Brocade 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 Brocade 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 Brocade device

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

```
Brocade(config)#web-management https
```

Syntax: **[no] web-management** [**http** | **https**]

You can enable either the HTTP or HTTPs servers with this command. You can disable both the HTTP and HTTPs servers by entering the following command.

```
Brocade(config)#no web-management
```

Syntax: **no web-management**

Specifying a port for SSL communication

By default, SSL protocol exchanges occur on TCP port 443. You can optionally change the port number used for SSL communication.

For example, the following command causes the device to use TCP port 334 for SSL communication.

```
Brocade(config)#ip ssl port 334
```

Syntax: [no] ip ssl port *port-number*

The default port for SSL communication is 443.

Changing the SSL server certificate key size

The default key size for Brocade-issued and imported digital certificates is 1024 bits. If desired, you can change the default key size to a value of 512, 2048, or 4096 bits. To do so, enter a command such as the following at the Global CONFIG level of the CLI.

```
Brocade(config)#ip ssl cert-key-size 512
```

Syntax: ip ssl cert-key-size *<512/ 1024/ 2048/ 4096>*

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

Support for SSL digital certificates larger than 2048 bits

Brocade 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 other Brocade device using an SSL connection, you configure a set of digital certificates and RSA public-private key pairs on the device. A digital certificate is used for identifying the connecting client to the server. It contains information about the issuing Certificate Authority, as well as a public key. You can either import digital certificates and private keys from a server, or you can allow the Brocade device to create them.

If you want to allow the Brocade device to create the digital certificates, refer to the next section, Generating an SSL certificate on page 82. If you choose to import an RSA certificate and private key file from a client, you can use TFTP to transfer the files.
For example, to import a digital certificate using TFTP, enter a command such as the following:

```
Brocade(config)#ip ssl certificate-data-file tftp 192.168.9.210 certfile
```

Syntax: [no] ip ssl certificate-data-file tftp*ip-address certificate-filename*

To import an RSA private key from a client using TFTP, enter a command such as the following:

```
Brocade(config)#ip ssl private-key-file tftp 192.168.9.210 keyfile
```

Syntax: [no] ip ssl private-key-file tftp *ip-address key-filename*

The *ip-address* is the IP address of a TFTP server that contains the digital certificate or private key.

> **NOTE**
> The RSA key can be up to 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.

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

Syntax: [no] crypto-ssl certificate generate

Deleting the SSL certificate

To delete the SSL certificate, enter the following command.

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

Syntax: [no] 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 which acts 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 which act as a 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 30 or Remote access restrictions on page 32.

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.
- 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 will authenticate access attempts using the locally configured usernames and passwords.

The command syntax for each of the following examples is provided in the *Command Syntax* section.

Example 1

To configure an authentication-method list for the Web Management Interface, enter a command such as the following.

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

This command configures the device to use the local user accounts to authenticate access to the device through the Web Management Interface. If the device does not have a user account that matches the user name and password entered by the user, the user is not granted access.

Example 2

To configure an authentication-method list for SNMP, enter a command such as the following.

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

This command allows certain incoming SNMP SET operations to be authenticated using the locally configured usernames and passwords. When this 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 need additional validation. These objects include but are not limited to: **snAgReload**, **snAgWriteNVRAM**, **snAgConfigFromNVRAM**, **snAgImgLoad**, **snAgCfgLoad** and **snAgGblTelnetPassword**. For more information, see **snAgGblPassword** in the MIB Reference Guide.

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

Note that the above 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 following command.

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

This command configures the device to use the local user accounts to authenticate attempts to access the Privileged EXEC and CONFIG levels of the CLI.

Example 4

To configure the device to consult a RADIUS server first to authenticate attempts to access the Privileged EXEC and CONFIG levels of the CLI, then consult the local user accounts if the RADIUS server is unavailable, enter the following command.

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

Command Syntax

The following is the command syntax for the preceding examples.

Syntax: [no] aaa authentication { **snmp-server** | **web-server** | **enable** | **login default** } *method 1* [*method 2-7*]

The **snmp-server** | **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**
> TACACS/TACACS+ and RADIUS are supported only with the **enable** and **login** parameters.

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 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 20.
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 59.
none	Do not use any authentication method. The device automatically permits access.

TCP Flags - edge port security

The edge port security feature works in combination with IP ACL rules, and supports all 6 TCP flags present in the offset 13 of the TCP header:

- +|- urg = Urgent
- +|- ack = Acknowledge
- +|- psh = Push
- +|- rst = Reset
- +|- syn = Synchronize
- +|- fin = Finish

TCP flags can be combined with other ACL functions (such as dscp-marking and traffic policies), giving you greater flexibility when designing ACLs.

The TCP flags feature offers two options, match-all and match-any:

- **Match-any** - Indicates that incoming TCP traffic must be matched against any of the TCP flags configured as part of the match-any ACL rule. In CAM hardware, the number of ACL rules will match the number of configured flags.
- **Match-all** - Indicates that incoming TCP traffic must be matched against all of the TCP flags configured as part of the match-all ACL rule. In Content-addressable memory (CAM) hardware, there will be only one ACL rule for all configured flags.

  NOTE
  The **match-all** option is not supported on ICX 7750 and ICX 7450 devices.

```
device(config-ext-nACL)#permit tcp 10.1.1.1 0.0.0.255 eq 100 10.2.2.2 0.0.0.255 eq 300 match-all +urg +ack
+syn -rst
```

This command configures a single rule in CAM hardware. This rule will contain all of the configured TCP flags (urg, ack, syn, and rst).

Using TCP Flags in combination with other ACL features

The TCP Flags feature has the added capability of being combined with other ACL features.

```
device(config-ext-nACL)#permit tcp any any match-all +urg +ack +syn -rst traffic-policy test
```

This command configures the ACL to match incoming traffic with the TCP Flags urg, ack, and syn and also to apply the traffic policy (rate, limit, etc.) to the matched traffic.

```
device(config-ext-nACL)#permit tcp any any match-all +urg +ack +syn -rst tos normal
```

This command configures the ACL to match incoming traffic with the flags urg, ack, and syn, and also sets the tos bit to normal when the traffic exits the device.

NOTE
TCP Flags combines the functionality of older features such as TCP Syn Attack and TCP Establish. Avoid configuring these older features on a port where you have configured TCP Flags. TCP Flags can perform all of the functions of TCP Syn Attack and TCP Establish, and more. However, if TCP Syn Attack is configured on a port along with TCP Flags, TCP Syn Attack will take precedence.

NOTE
If an ACL clause with match-any exists, and the system runs out of CAM, if the total number of TCP rules to TCP Flags will not fit within 1021 entries (the maximum rules allowed per device), then none of the TCP Flag rules will be programmed into the CAM hardware.

NOTE
If a range option and match-any TCP-flags are combined in the same ACL, the total number of rules will be calculated as: Total number of rules in CAM hardware = (number of rules for range)* (number of rules for match-any TCP-flags).

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 Brocade device negotiates the version of SSH2 to be used. The highest version of SSH2 supported by both the Brocade 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.

Brocade devices also support Secure Copy (SCP) for securely transferring files between a Brocade 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
- Brocade FastIron 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-group1-sha1** and **diffie-hellman-group14-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 91.
2. Configure DSA or RSA challenge-response authentication.

   See the section Configuring DSA or RSA challenge-response authentication on page 93.
3. Set optional parameters.

   See the section Optional SSH parameters on page 95.

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 Brocade 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
```

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

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

Syntax: crypto key { **generate** | **zeroize** } **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 **zeroize** keyword deletes the host key pair from the flash memory. This disables SSH if no other server host keys exist on the device.

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, and the command **crypto key zeroize** works as described in Deleting DSA and RSA key pairs on page 92.

Generating and deleting an RSA key pair

To generate an RSA key pair, enter a command such as the following:

```
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
```

Syntax: crypto key { **generate** | **zeroize** } **rsa** [**modulus** *modulus-size*]

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 optional [**modulus** *modulus-size*] parameter specifies the modulus size of the RSA key pair, in bits. The valid values for **modulus-size** are 1024 or 2048. The default value is 1024.

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.

The **rsa** keyword specifies an RSA host key pair.

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
```

Syntax: 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 Brocade 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 Brocade device in it.

If you are using SSH to connect to a Brocade device from a UNIX system, you may need to add the public key on the Brocade 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 Brocade 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 Ruckus 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 Brocade 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 in this 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 Brocade 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 a command such as the following:

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

Syntax: ip ssh pub-key-file { **tftp** *tftp-server-ip-addr filename* | **remove** }

The *tftp-server-ip-addr* variable is the IP address of the tftp server that contains the public key file that you want to import into the Ruckus device.

The *filename* variable is the name of the public key file that you want to import into the Ruckus device.

The **remove** parameter deletes the public keys from the device.

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 ----
```

Syntax: show ip client-pub-key [**begin** *expression* | **exclude** *expression* | **include** *expression*]

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

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

Syntax: clear public-key

Enabling DSA or RSA challenge-response authentication

DSA and RSA challenge-response authentication is enabled by default. You can disable or re-enable it manually.

To enable DSA and RSA challenge-response authentication.

```
device(config)#ip ssh password-authentication yes
```

To disable DSA and RSA challenge-response authentication.

```
device(config)#ip ssh password-authentication no
```

Syntax: ip ssh password-authentication{ **yes** | **no** }

To enable keyboard-interactive authentication:

```
device(config)#ip ssh interactive-authentication yes
```

To disable keyboard interactive authentication:

```
device(config)#ip ssh interactive-authentication no
```

Syntax: ip ssh interactive--authentication{ **yes** | **no** }

To enable public key authentication:

```
device(config)#ip ssh key-authentication yes
```

To disable public key authentication:

```
device(config)#ip ssh key-authentication no
```

Syntax: ip ssh interactive--authentication { **yes** | **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
- 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

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 Brocade devices which acts as an SSH client. When the Brocade 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
```

Syntax: ip ssh interactive--authentication-retries *number*

Deactivating user authentication

After the SSH server on the Ruckus device negotiates a session key and encryption method with the connecting client, user authentication takes place. The Ruckus implementation of SSH supports DSA or RSA challenge-response authentication and password 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.

With password authentication, users are prompted for a password when they attempt to log into the device (provided empty password logins are not allowed). If there is no user account that matches the user name and password supplied by the user, the user is not granted access.

You can deactivate one or both user authentication methods for SSH. Note that deactivating both authentication methods essentially disables the SSH server entirely.

To disable DSA or RSA challenge-response authentication, enter the following command.

```
device(config)#ip ssh key-authentication no
```

Syntax: ip ssh key--authentication { **yes** | **no** }

The default is **yes** .

To deactivate password authentication, enter the following command.

```
device(config)#ip ssh password-authentication no
```

Syntax: ip ssh password--authentication { **no** | **yes** }

The default is **yes** .

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(config)#ip ssh permit-empty-passwd yes
```

Syntax: ip ssh permit-empty-passwd { **no** | **yes** }

Setting the SSH port number

By default, SSH traffic occurs on TCP port 22. You can change this port number. For example, the following command changes the SSH port number to 2200.

```
device(config)#ip ssh port 2200
```

Note that if you change the default SSH port number, you must configure SSH clients to connect to the new port. Also, you should be careful not to assign SSH to a port that is used by another service. If you change the SSH port number, Ruckus recommends that you change it to a port number greater than 1024.

Syntax: ip ssh port *number*

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 between 1 - 120 seconds. For example, to change the timeout value to 60 seconds, enter the following command.

```
device(config)#ip ssh timeout 60
```

Syntax: ip ssh timeout *seconds*

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. Optionally, you can 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(config)#ip ssh idle-time 30
```

Syntax: ip ssh idle-time *minutes*

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.

Filtering SSH access using ACLs

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

Enter commands such as the following.

```
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
```

Syntax: ssh access-group { *standard-named-acl* | *standard-numbered-acl* }

Terminating an active SSH connection

To terminate one of the active SSH connections, enter the following command

```
device#kill ssh 1
```

Syntax: kill ssh *connection-id*

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)
```

Syntax: show ip ssh [**begin** *expression* | **exclude** *expression* | **include** *expression*]

This display shows the following information about the active SSH connections.

TABLE 13 SSH connection information

Field	Description
Inbound	Connections listed under this heading are inbound.
Outbound	Connections listed under this heading are outbound.
Connection	The SSH connection ID.
Version	The SSH version number.
Encryption	The encryption method used for the connection.
Username	The user name for the connection.
HMAC	The HMAC version
Server Hostkey	The type of server hostkey. This can be DSA or RSA.
IP Address	The IP address of the SSH client
SSH-v2.0 enabled	Indicates that SSHv2 is enabled.
hostkey	Indicates that at least one host key is on the device. It is followed by a list of the the host key types and modulus sizes.

Displaying SSH configuration information

To display SSH configuration information, use the **show ip ssh config** command:

```
Brocade# show ip ssh config
SSH server                  : Disabled
SSH port                    : tcp\22
Host Key                    : DSA 1024
Encryption                  : aes256-cbc, aes192-cbc, aes128-cbc, aes256-ctr, aes192-ctr, aes128-ctr, 3des-
cbc
Permit empty password       : Yes
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             :
Brocade#
```

Syntax: show ip ssh config

This display shows the following information.

Field	Description
SSH server	SSH server is enabled or disabled
SSH port	SSH port number
Encryption	The encryption used for the SSH connection. The following values are displayed when Standard mode is enabled: • aes256-ctr, aes192-ctr, aes128-ctr, aes256-cbc, aes192-cbc, aes128-cbc, 3des-cbc indicate the different AES methods used for encryption. • 3-DES indicates 3-DES algorithm is used for encryption.
Permit empty password	Empty password login is allowed or not allowed.
Authentication methods	The authentication methods used for SSH. The authentication can have one or more of the following values: • Password - indicates that you are prompted for a password when attempting to log into the device. • Public-key - indicates that DSA or RSA challenge-response authentication is enabled. • Interactive - indicates the interactive authentication si enabled.
Authentication retries	The number of authentication retries. This number can be from 1 to 5.
Login timeout (seconds)	SSH login timeout value in seconds. This can be from 0 to 120.
Idle timeout (minutes)	SSH idle timeout value in minutes. This can be from 0 to 240.
Strict management VRF	Strict management VRF is enabled or disabled.
SCP	SCP is enabled or disabled.
SSH IPv4 clients	The list of IPv4 addresses to which SSH access is allowed. The default is "All".
SSH IPv6 clients	The list of IPv4 addresses to which SSh access is allowed. Default "All".
SSH IPv4 access-list	The IPv4 ACL used to permit or deny access using SSH.
SSH IPv6 access-list	The IPv6 ACL used to permit or deny access to device using SSH.

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
```

Syntax: show who { **begin** *expression* | **exclude** *expression* | **include** *expression* }

SSH2 client

SSH2 client allows you to connect from a Brocade device to an SSH2 server, including another Brocade 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). Brocade 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 90.

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 on page 101

- Generating and deleting a client RSA key pair on page 101
- Exporting client public keys on page 101

Generating and deleting a client DSA key pair

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

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

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

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

Syntax: crypto key client { **generate** | **zeroize** } **dsa**

The **generate** keyword places a host key pair in the flash memory.

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

The **dsa** keyword specifies a DSA host key pair.

Generating and deleting a client RSA key pair

To generate a client RSA key pair, enter a command such as the following:

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

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

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

Syntax: crypto key client { **generate** | **zeroize** } **rsa** [**modulus** *modulus-size*]

The **generate** keyword places an RSA host key pair in the flash memory.

The **zeroize** keyword deletes the RSA host key pair from the flash memory.

The optional [**modulus** *modulus-size*] parameter specifies the modulus size of the RSA key pair, in bits. The valid values for *modulus-size* are 1024 or 2048. It is used only with the **generate** parameter. The default value is 1024.

The **rsa** keyword specifies an RSA host key pair.

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 brocade device, see the section Importing authorized public keys into the Ruckus device on page 93.

Using SSH2 client

To start an SSH2 client connection to an SSH2 server using password authentication, enter a command such as the following:

```
device# ssh 10.10.10.2
```

To start an SSH2 client connection to an SSH2 server using public key authentication, enter a command such as the following:

```
device# ssh 10.10.10.2 public-key dsa
```

Syntax: ssh *ipv4Addr* | *ipv6Addr* | *host-name* [**public-key** [**dsa** | **rsa**]] [**port** *portnum*]

The *ipv4Addr* , *ipv6Addr* , and *host-name* variables identify an SSH2 server. You identify the server to connect to by entering its IPv4 or IPv6 address or its hostname.

The optional [**public-key** [**dsa** | **rsa**]] parameter specifies the type of public key authentication to use for the connection, either DSA or RSA. If you do not enter this parameter, the default authentication type is password.

The optional **port** *portnum* parameter specifies that the SSH2 connection will use a non-default SSH2 port, where *portnum* is the port number. The default port number is 22.

Displaying SSH2 client information

For information about displaying SSH2 client information, see the following sections:

- Displaying SSH connection information on page 98
- Displaying additional SSH connection information on page 99

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 Brocade FastIron 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 Brocade 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.

NOTE
The SCP file transfer speed enhancement is supported only on Brocade ICX 7750, Brocade ICX 7450, and Brocade ICX 7250.

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.

```
Device#copy scp flash 10.20.1.1 SPR08040.bin primary
Device#copy scp flash 10.20.1.1 SPR08040.bin secondary
```

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. To disable SCP, enter the following command.

```
device(config)#ip ssh scp disable
```

Syntax: ip ssh [**scp**] { **disable** | **enable** }

NOTE
If you disable SSH, SCP is also disabled.

Secure copy configuration notes

- When using SCP, enter the **scp** commands on the SCP-enabled client, rather than the console on the Ruckus device.
- 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 (c:\cfg\brocade.cfg) to the running configuration file on a Brocade device at 10.168.1.50 and log in as user terry, enter the following command on the SCP-enabled client.

```
C:\> scp c:\cfg\brocade.cfg terry@10.168.1.50:runConfig
```

If password authentication is enabled for SSH, the user is prompted for user terry password before the file transfer takes place.

Copying a file to the startup config

To copy the configuration file to the startup configuration file, enter the following command.

```
C:\> scp c:\cfg\brocade.cfg terry@10.168.1.50:startConfig
```

Copying the running config file to an SCP-enabled client

To copy the running configuration file on the Brocade device to a file called c:\cfg\fdryrun.cfg on the SCP-enabled client, enter the following command.

```
C:\> scp terry@10.168.1.50:runConfig c:\cfg\brcdrun.cfg
```

Copying the startup config file to an SCP-enabled client

To copy the startup configuration file on the Ruckus device to a file called c:\cfg\brcdestart.cfg on the SCP-enabled client, enter the following command.

```
C:\> scp terry@10.168.1.50:startConfig c:\cfg\brcdstart.cfg
```

To overwrite the running configuration file

```
C:\> scp c:\cfg\brocade.cfg terry@10.168.1.50:runConfig-overwrite
```

Copying a software image file to flash memory

The **scp** command syntax differs between device series. Use the command syntax in the appropriate section.

Brocade ICX Devices

To copy a software image file from an SCP-enabled client to the primary flash on these devices, enter one of the following commands.

```
C:\> scp SPR08040.bin terry@10.168.1.50:flash:primary
```

or

```
C:\>scp SPR08040.bin terry@10.168.1.50:flash:pri:SPR08040.bin
```

To copy a software image file from an SCP-enabled client to the secondary flash on these devices, enter one of the following commands.

```
C:\> scp SPR08040.bin terry@10.168.1.50:flash:secondary
```

or

```
c:\> scp SPR08040.bin terry@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 will show the partition (primary or secondary) as EMPTY.

NOTE
The Brocade device supports only one SCP copy session at a time.

Copying a Software Image file from flash memory

The **scp** command syntax differs between device series. Use the command syntax in the appropriate section.

To copy a software image file from the primary flash on these devices to an SCP-enabled client, enter a command such as the following.

```
C:\> scp terry@10.168.1.50:flash:primary
 SPR08040.bin
```

To copy a software image file from the secondary flash on these devices to an SCP-enabled client, enter a command such as the following.

```
C:\> scp terry@10.168.1.50:flash:secondary
 SPR08040.bin
```

Importing a digital certificate using SCP

To import a digital certificate using SCP, enter a command such as the following one:

```
C:\> scp certfile user@10.168.89.210:sslCert
```

Syntax: scp *certificate-filenameuser@ip-address* **:sslCert**

The *ip-address* variable is the IP address of the server from which the digital certificate file is downloaded.

The *certificate-filename* variable is the file name of the digital certificate that you are importing to the device.

The **scp** command can be used when TFTP access is unavailable or not permitted and the command has an equivalent functionality to the **ip ssl certificate-data-file tftp** .

Importing an RSA private key

To import an RSA private key from a client using SCP, enter a command such as the following one:

```
C:\> scp keyfile user@10.168.9.210:sslPrivKey
```

Syntax: scp *key-filenameuser@ip-address* **sslPrivKey**

The *ip-address* variable is the IP address of the server that contains the private key file.

The *key-filename* variable is the file name of the private key that you want to import into the device.

The **scp** command can be used when TFTP access is unavailable or not permitted and the command has an equivalent functionality to the **ip ssl private-key-file tftp** command.

Importing a DSA or RSA public key

To import a DSA or RSA public key from a client using SCP, enter a command such as the following one:

```
C:\> scp pkeys.txt user@10.168.1.234:sshPubKey
```

Syntax: scp *key-filenameuser@ip-address* **:sshPubKey**

The *ip-address* variable is the IP address of the server that contains the public key file.

The *key-filename* variable is the name of the DSA or RSA public key file that you want to import into the device.

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 Ruckus device on page 93.

Copying license files

To copy the license files from a client using SCP, enter commands such as the following:

For stacking products:

```
C:\> scp license.xml user@10.168.1.234:license:3  (unit3)
```

Syntax: scp *license-filenameuser@ip-address* **:license**

IP ACLs

ACL overview

Ruckus devices support **rule-based ACLs** (sometimes called hardware-based ACLs), where the decisions to permit or deny packets are processed in hardware and all permitted packets are switched or routed in hardware. All denied packets are also dropped in hardware. The ACL features supported on inbound and outbound traffic are discussed in more detail in the rest of this chapter.

NOTE
FastIron devices do not support flow-based ACLs.

Rule-based ACLs program the ACL entries you assign to an interface into Content Addressable Memory (CAM) space allocated for the ports. The ACLs are programmed into hardware at startup (or as new ACLs are entered and bound to ports). Devices that use rule-based ACLs program the ACLs into the CAM entries and use these entries to permit or deny packets in the hardware, without sending the packets to the CPU for processing.

Rule-based ACLs are supported on the following interface types:

- Gbps Ethernet ports
- 10 Gbps Ethernet ports
- Trunk groups
- Virtual routing interfaces

Types of IP ACLs

You can configure the following types of IP ACLs:

- Standard - Permits or denies packets based on source IP address. Valid standard ACL IDs are 1 - 99 or a character string.
- Extended - Permits or denies packets based on source and destination IP address and also based on IP protocol information. Valid extended ACL IDs are a number from 100 - 199 or a character string.

ACL IDs and entries

ACLs consist of ACL IDs and ACL entries:

- ACL ID - An ACL ID is a number from 1 - 99 (for a standard ACL) or 100 - 199 (for an extended ACL) or a character string. The ACL ID identifies a collection of individual ACL entries. When you apply ACL entries to an interface, you do so by applying the ACL ID that contains the ACL entries to the interface, instead of applying the individual entries to the interface. This makes applying large groups of access filters (ACL entries) to interfaces simple. Refer to Numbered and named ACLs on page 112.

  NOTE
  This is different from IP access policies. If you use IP access policies, you apply the individual policies to interfaces.

- ACL entry - Also called an ACL rule, this is a filter command associated with an ACL ID. The maximum number of ACL rules you can configure is a system-wide parameter and depends on the device you are configuring. You can configure up to the maximum number of rules in any combination of different ACLs. The total number of rules in all ACLs cannot exceed the system maximum listed in the following table.

TABLE 14 Maximum number of ACL rules

Switch	Maximum ACL rules per port region	Maximum ACL rules per system (stand-alone switch or stack of switches)
ICX 7250	2815	8192
ICX 7450	3071	8192
ICX 7750	2047	8192

You configure ACLs on a global basis, then apply them to the incoming or outgoing traffic on specific ports. The software applies the rules within an ACL in the order they appear in the ACL configuration. As soon as a match is found, the software takes the action specified in the ACL rule (permit or deny the packet) and stops further comparison for that packet.

Numbered and named ACLs

When you configure an ACL, you can refer to the ACL by a numeric ID or by an alphanumeric name. The commands to configure numbered ACLs are different from the commands for named ACLs.

- Numbered ACL - If you refer to the ACL by a numeric ID, you can use 1 - 99 for a standard ACL or 100 - 199 for an extended ACL.
- Named ACL - If you refer to the ACL by a name, you specify whether the ACL is a standard ACL or an extended ACL, then specify the name.

You can configure up to 99 standard numbered IP ACLs and 100 extended numbered IP ACLs. You also can configure up to 99 standard named ACLs and 100 extended named ACLs.

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:

- 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 might want to configure ACLs that consist of explicit deny entries, 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.

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, since the default action on an interface that has ACLs is to deny the packet.

How fragmented packets are processed

The default processing of fragments by hardware-based ACLs is as follows:

- 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, since 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.
- For other fragments of the same packet, they 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, since 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 137.

Hardware aging of Layer 4 CAM entries

Rule-based ACLs use Layer 4 CAM entries. The device permanently programs rule-based ACLs into the CAM. The entries never age out.

ACL configuration considerations

- Hardware-based ACLs are supported on the following interface types:
  - Gbps Ethernet ports

  - 10 Gbps Ethernet ports
  - Trunk groups
  - Virtual routing interfaces

  NOTE
  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. This must be enabled using the **enable egress-acl-on-control-traffic** command. See Applying egress ACLs to Control (CPU) traffic on page 131 for details.
- The number of ACLs supported per device is listed in the *Maximum number of ACL entries* table.
- Hardware-based ACLs support only one ACL per port. The ACL of course 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.
- Inbound ACLs and outbound ACLs can co-exist. 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 the Ruckus 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, since 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. Ruckus ports do not support IP source guard and ACLs 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) If 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-AUTH/DOT1X enabled, as they the two are configured in different directions.
- The following ACL features and options are not supported on the FastIron devices:
  - Applying an ACL on a device that has Super Aggregated VLANs (SAVs) enabled.
  - ACL logging of permitted packets- ACL logging is supported for packets that are sent to the CPU for processing (denied packets) for inbound traffic. ACL logging is not supported for packets that are processed in hardware (permitted packets).
  - Flow-based ACLs
  - Layer 2 ACLs
- You can apply an ACL to a port that has TCP SYN protection or ICMP smurf protection, or both, enabled.

Configuring standard numbered ACLs

This section describes how to configure standard numbered ACLs with numeric IDs and provides configuration examples.

Standard ACLs permit or deny packets based on source IP address. You can configure up to 99 standard numbered ACLs. There is no limit to the number of ACL entries an ACL can contain except for the system-wide limitation. For the number of ACL entries supported on a device, refer to ACL IDs and entries on page 112.

Standard numbered ACL syntax

Syntax: [no] access-list *ACL-num* { **deny** | **permit** } { *source-ip* | *hostnamewildcard* } [**log**]

or

Syntax: [no] access-list *ACL-num* { **deny** | **permit** } { *source-ip/mask-bits* | *hostname* } [**log**]

Syntax: [no] access-list *ACL-num* { **deny** | **permit** } { *source-ip* | *hostname* } [**log**]

Syntax: [no] access-list *ACL-num* { **deny** | **permit** } **any** [**log**]

Syntax: [no] ip access-group *ACL-num* [**in** | **out**]

The *ACL-num* parameter is the access list number from 1 - 99.

The **deny | permit** parameter indicates whether packets that match a policy in the access list are denied (dropped) or permitted (forwarded).

The *source-ip* parameter specifies the source IP address. Alternatively, you can specify the host name.

> **NOTE**
> To specify the host name instead of the IP address, the host name must be configured using the DNS resolver on the Ruckus device. To configure the DNS resolver name, use the **ip dns server-address** ... command at the global CONFIG level of the CLI.

The *wildcard* parameter specifies the mask value to compare against the host address specified by the source-ip parameter. 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 to 255, for example 0.0.0.255. Zeros in the mask mean the packet source address must match the source-ip . Ones mean any value matches. For example, the source-ip and wildcard values 10.157.22.26 0.0.0.255 mean that all hosts in the Class C subnet 10.157.22.x match the policy.

If you prefer to specify the wildcard (mask value) in CIDR format, you can enter a forward slash after the IP address, then enter the number of significant bits in the mask. For example, you can enter the CIDR equivalent of "10.157.22.26 0.0.0.255" as "10.157.22.26/24". The CLI automatically converts the CIDR number into the appropriate ACL mask (where zeros instead of ones are the significant bits) and changes the non-significant portion of the IP address into ones. For example, if you specify 10.157.22.26/24 or 10.157.22.26 0.0.0.255, then save the changes to the startup-config file, the value appears as 10.157.22.0/24 (if you have enabled display of subnet lengths) or 10.157.22.0 0.0.0.255 in the startup-config file.

If you enable the software to display IP subnet masks in CIDR format, the mask is saved in the file in "/ mask-bits " format. To enable the software to display the CIDR masks, enter the **ip show-subnet-length** command at the global CONFIG level of the CLI. You can use the CIDR format to configure the ACL entry regardless of whether the software is configured to display the masks in CIDR format.

> **NOTE**
> If you use the CIDR format, the ACL entries appear in this format in the running-config and startup-config files, but are shown with subnet mask in the display produced by the **show ip access-list** command.

The **host** source-ip | hostname parameter lets you specify a host IP address or name. When you use this parameter, you do not need to specify the mask. A mask of all zeros (0.0.0.0) is implied.

The **any** parameter configures the policy to match on all host addresses.

The **log** argument configures the device to generate Syslog entries and SNMP traps for inbound packets that are denied by the access policy.

The **in | out** parameter applies the ACL to incoming or outgoing traffic on the interface to which you apply the ACL. You can apply the ACL to an Ethernet port, or virtual interface.

NOTE
If the ACL is for a virtual routing interface, you also can specify a subset of ports within the VLAN containing that interface when assigning an ACL to the interface.

Configuration example for standard numbered ACLs

To configure a standard ACL and apply it to incoming traffic on port 1/1/1, enter the following commands.

```
device(config)# access-list 1 deny host 10.157.22.26 log
device(config)# access-list 1 deny 10.157.29.12 log
device(config)# access-list 1 deny host IPHost1 log
device(config)# access-list 1 permit any
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)#ip access-group 1 in
device(config)# write memory
```

The commands in this example configure an ACL to deny packets from three source IP addresses from being received on port 1/1/1. The last ACL entry in this ACL permits all packets that are not explicitly denied by the first three ACL entries.

Standard named ACL configuration

This section describes how to configure standard named ACLs with alphanumeric IDs. This section also provides configuration examples.

Standard ACLs permit or deny packets based on source IP address. You can configure up to 99 standard named ACLs. There is no limit to the number of ACL entries an ACL can contain except for the system-wide limitation. For the number of ACL entries supported on a device, refer to ACL IDs and entries on page 112.

The commands for configuring named ACL entries are different from the commands for configuring numbered ACL entries. The command to configure a numbered ACL is **access-list** . The command for configuring a named ACL is **ip access-list** . In addition, when you configure a numbered ACL entry, you specify all the command parameters on the same command. When you configure a named ACL, you specify the ACL type (standard or extended) and the ACL name with one command, which places you in the configuration level for that ACL. Once you enter the configuration level for the ACL, the command syntax is the same as the syntax for numbered ACLs.

Standard named ACL syntax

Syntax: [no] ip access-list standard {*ACL-name* | *ACL-num* } { **deny** | **permit** } { *source-ip* | *hostname wildcard* } [**log**]

or

Syntax: [no] ip access-list standard {*ACL-name* | *ACL-num* } { **deny** | **permit** } { *source-ip/mask-bits* | *hostname* } [**log**]

Syntax: [no] ip access-list standard {*ACL-name* |*ACL-num* } { **deny** | **permit** } { *source-ip* | *hostname* } [**log**]

Syntax: [no] ip access-list standard {*ACL-name* |*ACL-num* } { { **deny** | **permit** } **any** [**log**]

Syntax: [no] ip access-group *ACL-name* [**in** | **out**]

The *ACL-name* parameter is the access list name. You can specify a string of up to 255 alphanumeric characters. You can use blanks in the ACL name if you enclose the name in quotation marks (for example, "ACL for Net1"). ACL names must be unique across all IPv4 and IPv6 ACLs.

The *ACL-num* parameter allows you to specify an ACL number if you prefer. If you specify a number, you can specify from 1 - 99 for standard ACLs.

NOTE
For convenience, the software allows you to configure numbered ACLs using the syntax for named ACLs. The software also still supports the older syntax for numbered ACLs. Although the software allows both methods for configuring numbered ACLs, numbered ACLs are always formatted in the startup-config and running-config files in using the older syntax, as follows. `access-list 1 deny host 10.157.22.26 logaccess-list 1 deny 10.157.22.0 0.0.0.255 logaccess-list 1 permit any access-list 101 deny tcp any any eq http log`

The **deny | permit** parameter indicates whether packets that match a policy in the access list are denied (dropped) or permitted (forwarded).

The *source-ip* parameter specifies the source IP address. Alternatively, you can specify the host name.

NOTE
To specify the host name instead of the IP address, the host name must be configured using the DNS resolver on the Ruckus device. To configure the DNS resolver name, use the **ip dns server-address** ... command at the global CONFIG level of the CLI.

The *wildcard* parameter specifies the mask value to compare against the host address specified by the source-ip parameter. 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 to 255, for example 0.0.0.255. Zeros in the mask mean the packet source address must match the source-ip . Ones mean any value matches. For example, the source-ip and wildcard values 10.157.22.26 0.0.0.255 mean that all hosts in the Class C subnet 10.157.22.x match the policy.

If you prefer to specify the wildcard (mask value) in CIDR format, you can enter a forward slash after the IP address, then enter the number of significant bits in the mask. For example, you can enter the CIDR equivalent of "10.157.22.26 0.0.0.255" as "10.157.22.26/24". The CLI automatically converts the CIDR number into the appropriate ACL mask (where zeros instead of ones are the significant bits) and changes the non-significant portion of the IP address into ones. For example, if you specify 10.157.22.26/24 or 10.157.22.26 0.0.0.255, then save the changes to the startup-config file, the value appears as 10.157.22.0/24 (if you have enabled display of subnet lengths) or 10.157.22.0 0.0.0.255 in the startup-config file.

If you enable the software to display IP subnet masks in CIDR format, the mask is saved in the file in "/ mask-bits " format. To enable the software to display the CIDR masks, enter the **ip show-subnet-length** command at the global CONFIG level of the CLI. You can use the CIDR format to configure the ACL entry regardless of whether the software is configured to display the masks in CIDR format.

NOTE
If you use the CIDR format, the ACL entries appear in this format in the running-config and startup-config files, but are shown with subnet mask in the display produced by the **show ip access-list** command.

The **host** *source-ip* | *hostname* parameter lets you specify a host IP address or name. When you use this parameter, you do not need to specify the mask. A mask of all zeros (0.0.0.0) is implied.

The **any** parameter configures the policy to match on all host addresses.

The **log** argument configures the device to generate Syslog entries and SNMP traps for inbound packets that are denied by the access policy.

NOTE
You can enable logging on inbound ACLs and filters that support logging even when the ACLs and filters are already in use. To do so, re-enter the ACL or filter command and add the **log** parameter to the end of the ACL or filter. The software replaces the ACL or filter command with the new one. The new ACL or filter, with logging enabled, takes effect immediately.

The **in | out** parameter applies the ACL to incoming or outgoing traffic on the interface to which you apply the ACL. You can apply the ACL to an Ethernet port or virtual interface.

> **NOTE**
> If the ACL is bound to a virtual routing interface, you also can specify a subset of ports within the VLAN containing that interface when assigning an ACL to the interface. See Enabling ACL filtering based on VLAN membership or VE port membership on page 138 for further details.

Configuration example for standard named ACLs

To configure a standard named ACL, enter commands such as the following.

```
device(config)# ip access-list standard Net1
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 1/1/1
device(config-if-e1000-1/1/1)# ip access-group Net1 in
```

The commands in this example configure a standard ACL named "Net1". The entries in this ACL deny packets from three source IP addresses from being forwarded on port 1. Since the implicit action for an ACL is "deny", the last ACL entry in this ACL permits all packets that are not explicitly denied by the first three ACL entries. For an example of how to configure the same entries in a numbered ACL, refer to Configuring standard numbered ACLs on page 114.

Notice that the command prompt changes after you enter the ACL type and name. The "std" in the command prompt indicates that you are configuring entries for a standard ACL. For an extended ACL, this part of the command prompt is "ext". The "nACL" indicates that you are configuring a named ACL.

Extended numbered ACL configuration

This section describes how to configure extended numbered ACLs.

Extended ACLs let you permit or deny packets based on the following information:

- IP protocol
- Source IP address or host name
- Destination IP address or host name
- Source TCP or UDP port (if the IP protocol is TCP or UDP)
- Destination TCP or UDP port (if the IP protocol is TCP or UDP)

The IP protocol can be one of the following well-known names or any IP protocol number from 0 - 255:

- Internet Control Message Protocol (ICMP)
- Internet Group Management Protocol (IGMP)
- Internet Gateway Routing Protocol (IGRP)
- Internet Protocol (IP)
- Open Shortest Path First (OSPF)
- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

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 IP address to the website IP address.

Extended numbered ACL syntax

Syntax: [no] access-list *ext-ACL-num* { **deny** | **permit** } *ip-protocol* { *source-ip* | *hostname wildcard* } [*operator* [*source-tcp* | *udp-port*]] | *destination-ip* | *hostname* [*icmp-num* | *icmp-type*] *wildcard* [**tcp** | **udp**] *comparison operator destination* [**tcp** | **udp** *port*] [**dscp-cos-mapping**] [**dscp-marking** *0-63* [**802.1p-priority-marking** *0-7...* | **dscp-cos-mapping**]] [**precedence** *name* | *0-7*] [**tos** *0-63* | *name*] [**dscp-matching** *0-63*] [**802.1p-priority-matching** *0-7*] [**log**] [**traffic-policy** *name*]

Syntax: [no] access-list *ACL-num* { **deny** | **permit** } **host** *ip-protocol* **any any**

Syntax: [no] ip access-group *ACL-num* [**in** | **out**]

The *ext-ACL-num* parameter is the extended access list number. Specify a number from 100 through 199.

The **deny** | **permit** parameter indicates whether packets that match the policy are dropped or forwarded.

The *ip-protocol* parameter indicates the type of IP packet you are filtering. You can specify a well-known name for any protocol whose number is less than 255. For other protocols, you must enter the number. Enter "?" instead of a protocol to list the well-known names recognized by the CLI.

The *source-ip* | *hostname* parameter specifies the source IP host for the policy. If you want the policy to match on all source addresses, enter **any** .

The *wildcard* parameter specifies the portion of the source IP host address to match against. 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 to 255, for example 0.0.0.255. Zeros in the mask mean the packet's source address must match the *source-ip* . Ones mean any value matches. For example, the source-ip and wildcard values 10.157.22.26 0.0.0.255 mean that all hosts in the Class C subnet 10.157.22.x match the policy.

If you prefer to specify the wildcard (mask value) in Classless Interdomain Routing (CIDR) format, you can enter a forward slash after the IP address, then enter the number of significant bits in the mask. For example, you can enter the CIDR equivalent of "10.157.22.26 0.0.0.255" as "10.157.22.26/24". The CLI automatically converts the CIDR number into the appropriate ACL mask (where zeros instead of ones are the significant bits) and changes the non-significant portion of the IP address into zeros. For example, if you specify 10.157.22.26/24 or 10.157.22.26 0.0.0.255, then save the changes to the startup-config file, the value appears as 10.157.22.0/24 (if you have enabled display of subnet lengths) or 10.157.22.0 0.0.0.255 in the startup-config file.

If you enable the software to display IP subnet masks in CIDR format, the mask is saved in the file in "/ mask-bits " format. To enable the software to display the CIDR masks, enter the **ip show-subnet-length** command at the global CONFIG level of the CLI. You can use the CIDR format to configure the ACL entry regardless of whether the software is configured to display the masks in CIDR format.

> **NOTE**
> If you use the CIDR format, the ACL entries appear in this format in the running-config and startup-config files, but are shown with subnet mask in the display produced by the **show ip access-list** command.

The *destination-ip* | *hostname* parameter specifies the destination IP host for the policy. If you want the policy to match on all destination addresses, enter **any**.

The *icmp-type* | *icmp-num* parameter specifies the ICMP protocol type:

- This parameter applies only if you specified **icmp** as the ip-protocol value.
- If you use this parameter, the ACL entry is sent to the CPU for processing.

- If you do not specify a message type, the ACL applies to all types of ICMP messages.

The *icmp-num* parameter can be a value from 0 through 255.

The *icmp-type* parameter can have one of the following values, depending on the software version the device is running:

- any-icmp-type
- echo
- echo-reply
- information-request
- log
- mask-reply
- mask-request
- parameter-problem
- redirect
- source-quench
- time-exceeded
- timestamp-reply
- timestamp-request
- traffic policy
- unreachable
- num

  NOTE
  The QoS options listed below are only available if a specific ICMP type is specified for the icmp-type parameter and cannot be used with the **any-icmp-type** option above. See QoS options for IP ACLs on page 143 for more information on using ACLs to perform QoS.

The tcp/udp comparison operator parameter specifies a comparison operator for the TCP or UDP port number. This parameter applies only when you specify **tcp** or **udp** as the IP protocol. For example, if you are configuring an entry for HTTP, specify **tcp eq http** . You can enter one of the following operators:

- **eq** - The policy applies to the TCP or UDP port name or number you enter after **eq** .
- **established** - This operator applies only to TCP packets. If you use this operator, the policy applies to TCP packets that have the ACK (Acknowledgment) or RST (Reset) bits set on (set to "1") in the Control Bits field of the TCP packet header. Thus, the policy applies only to established TCP sessions, not to new sessions. Refer to Section 3.1, "Header Format", in RFC 793 for information about this field.

  NOTE
  This operator applies only to destination TCP ports, not source TCP ports.

- **gt** - The policy applies to TCP or UDP port numbers greater than the port number or the numeric equivalent of the port name you enter after **gt** .
- **lt** - The policy applies to TCP or UDP port numbers that are less than the port number or the numeric equivalent of the port name you enter after **lt** .
- **neq** - The policy applies to all TCP or UDP port numbers except the port number or port name you enter after **neq** .
- **range** - The policy applies to all TCP or UDP port numbers that are between the first TCP or UDP port name or number and the second one you enter following the **range** parameter. The range includes the port names or numbers you enter.

For example, to apply the policy to all ports between and including 23 (Telnet) and 53 (DNS), enter the following: **range 23 53** . The first port number in the range must be lower than the last number in the range.

The *tcp/udp-port* parameter specifies the TCP or UDP port number or well-known name. You can specify a well-known name for any application port whose number is less than 1024. For other application ports, you must enter the number. Enter "?" instead of a port to list the well-known names recognized by the CLI.

The **in** | **out** parameter specifies that the ACL applies to incoming traffic on the interface to which you apply the ACL. You can apply the ACL to an Ethernet port or a virtual interface.

NOTE
If the ACL is for a virtual routing interface, you also can specify a subset of ports within the VLAN containing that interface when assigning an ACL to the interface. Refer to Configuring standard numbered ACLs on page 114.

The **precedence** *name* | *num* parameter specifies the IP precedence. The precedence option for an IP packet is set in a three-bit field following the four-bit header-length field of the packet's header. You can specify one of the following:

- **critical** or **5** - The ACL matches packets that have the critical precedence. If you specify the option number instead of the name, specify number 5.
- **flash** or **3** - The ACL matches packets that have the flash precedence. If you specify the option number instead of the name, specify number 3.
- **flash-override** or **4** - The ACL matches packets that have the flash override precedence. If you specify the option number instead of the name, specify number 4.
- **immediate** or **2** - The ACL matches packets that have the immediate precedence. If you specify the option number instead of the name, specify number 2.
- **internet** or **6** - The ACL matches packets that have the internetwork control precedence. If you specify the option number instead of the name, specify number 6.
- **network** or **7** - The ACL matches packets that have the network control precedence. If you specify the option number instead of the name, specify number 7.
- **priority** or **1** - The ACL matches packets that have the priority precedence. If you specify the option number instead of the name, specify number 1.
- **routine** or **0** - The ACL matches packets that have the routine precedence. If you specify the option number instead of the name, specify number 0.

The **tos** *name* | *num* parameter of the **ip access-list** command specifies the IP ToS. You can specify one of the following:

- **max-reliability** or **2** - The ACL matches packets that have the maximum reliability ToS. The decimal value for this option is 2.
- **max-throughput** or **4** - The ACL matches packets that have the maximum throughput ToS. The decimal value for this option is 4.
- **min-delay** or **8** - The ACL matches packets that have the minimum delay ToS. The decimal value for this option is 8.
- **min-monetary-cost** or **1** - The ACL matches packets that have the minimum monetary cost ToS. The decimal value for this option is 1.

  NOTE
  This value is not supported on 10 Gigabit Ethernet modules.

- – **normal** or **0** - The ACL matches packets that have the normal ToS. The decimal value for this option is 0.
  – *num* - A number from 0 - 15 that is the sum of the numeric values of the options you want. The ToS field is a four-bit field following the Precedence field in the IP header. You can specify one or more of the following. To select more than one option, enter the decimal value that is equivalent to the sum of the numeric values of all the ToS options

you want to select. For example, to select the **max-reliability** and **min-delay** options, enter number 10. To select all options, select 15.

NOTE
The following QoS options are only available if a specific ICMP type is specified and cannot be used with the **any-icmp-type** option set for the icmp-type parameter. See QoS options for IP ACLs on page 143 for more information on using ACLs to perform QoS.

The **802.1p-priority-matching** option inspects the 802.1p bit in the ACL that can be used with adaptive rate limiting. Enter a value from 0 - 7. For details, refer to "Inspecting the 802.1p bit in the ACL for adaptive rate limiting" section in the *Brocade FastIron Traffic Management Configuration Guide***.**

The **dscp-marking** option enables you to configure an ACL that marks matching packets with a specified DSCP value. Enter a value from 0 - 63.

The **dscp-matching** option matches on the packet's DSCP value. Enter a value from 0 - 63. This option does not change the packet's forwarding priority through the device or mark the packet. Refer to DSCP matching on page 144.

The **log** parameter enables SNMP traps and Syslog messages for inbound packets denied by the ACL:

- You can enable logging on inbound ACLs and filters that support logging even when the ACLs and filters are already in use. To do so, re-enter the **ACL** or **filter** command and add the log parameter to the end of the ACL or filter. The software replaces the **ACL** or **filter** command with the new one. The new ACL or filter, with logging enabled, takes effect immediately.

The **traffic-policy** option enables the device to rate limit inbound traffic and to count the packets and bytes per packet to which ACL permit or deny clauses are applied. For configuration procedures and examples, refer to the chapter "Traffic Policies" in the *Brocade FastIron Traffic Management Configuration Guide*.

To configure an extended named ACL, enter the **ip access-list extended** command.

```
device(config)#ip access-list extended "block Telnet"
device(config-ext-nACL)#deny tcp host 10.157.22.26 any eq telnet log
device(config-ext-nACL)#permit ip any any
device(config-ext-nACL)#exit
device(config)#interface ethernet 1/1/1
device(config-if-1/1/1)#ip access-group "block Telnet" in
```

The options at the ACL configuration level and the syntax for the **ip access-group** command are the same for numbered and named ACLs and are described in Extended numbered ACL configuration on page 118 and Extended numbered ACL configuration on page 118.

Configuration examples for extended numbered ACLs

To configure an extended access control list that blocks all Telnet traffic received on port 1/1/1 from IP host 10.157.22.26, enter the following commands.

```
device(config)#access-list 101 deny tcp host 10.157.22.26 any eq telnet log
device(config)#access-list 101 permit ip any any
device(config-if-e1000-1/1/1)#ip access-group 101 in
device(config)#interface ethernet 1/1/1
device(config-if-e1000-1/1/1)#exit
device(config)#write memory
```

Here is another example of commands for configuring an extended ACL and applying it to an interface. These examples show many of the syntax choices. Notice that some of the entries are configured to generate log entries while other entries are not thus configured.

```
device(config)#access-list 102 perm icmp 10.157.22.0/24 10.157.21.0/24
device(config)#access-list 102 deny igmp host rkwong 10.157.21.0/24 log
device(config)#access-list 102 deny igrp 10.157.21.0/24 host rkwong log
device(config)#access-list 102 deny ip host 10.157.21.100 host 10.157.22.1 log
device(config)#access-list 102 deny ospf any any log
device(config)#access-list 102 permit ip any any
```

The first entry permits ICMP traffic from hosts in the 10.157.22.x network to hosts in the 10.157.21.x network.

The second entry denies IGMP traffic from the host device named "rkwong" to the 10.157.21.x network.

The third entry denies IGMP traffic from the 10.157.21.x network to the host device named "rkwong".

The fourth entry denies all IP traffic from host 10.157.21.100 to host 10.157.22.1 and generates Syslog entries for packets that are denied by this entry.

The fifth entry denies all OSPF traffic and generates Syslog entries for denied traffic.

The sixth 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 commands apply ACL 102 to the incoming traffic on port 1/1/2 and to the incoming traffic on port 1/4/3.

```
device(config)# interface ethernet 1/1/2
        device(config-if-1/1/2)# ip access-group 102 in
device(config-if-1/1/2)# exit
device(config)# interface ethernet 1/4/3
device(config-if-1/4/3)# ip access-group 102 in
device(config-if-1/4/3)#exit
device(config)# write memory
```

Here is another example of an extended ACL.

```
device(config)#access-list 103 deny tcp 10.157.21.0/24 10.157.22.0/24
device(config)#access-list 103 deny tcp 10.157.21.0/24 eq ftp 10.157.22.0/24
device(config)#access-list 103 deny tcp 10.157.21.0/24 10.157.22.0/24 lt
         telnet neq 5
device(config)#access-list 103 deny udp any range 5 6 10.157.22.0/24 range 7 8
device(config)#access-list 103 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.

The second entry denies all FTP traffic from the 10.157.21.x network to the 10.157.22.x network.

The third entry denies TCP traffic from the 10.157.21.x network to the 10.157.22.x network, if the TCP port number of the traffic is less than the well-known TCP port number for Telnet (23), and if the TCP port is not equal to 5. Thus, TCP packets whose TCP port numbers are 5 or are greater than 23 are allowed.

The fourth entry denies UDP packets from any source to the 10.157.22.x network, if the UDP port number from the source network is 5 or 6 and the destination UDP port is 7 or 8.

The fifth 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 commands apply ACL 103 to the incoming traffic on ports 1/2/1 and 1/2/2.

```
device(config)# interface ethernet 1/2/1
device(config-if-1/2/1)# ip access-group 103 in
device(config-if-1/2/1)# exit
device(config)# interface ethernet 1/2/2
device(config-if-1/2/2)# ip access-group 103 in
```

```
device(config-if-1/4/3)#exit
device(config)#write memory
```

Extended ACL definition

The **deny** | **permit** parameter indicates whether packets that match the policy are dropped or forwarded.

The *ip-protocol* parameter indicates the type of IP packet you are filtering. You can specify a well-known name for any protocol whose number is less than 255. For other protocols, you must enter the number. Enter "?" instead of a protocol to list the well-known names recognized by the CLI.

The *source-ip* | *hostname* parameter specifies the source IP host for the policy. If you want the policy to match on all source addresses, enter **any** .

The *wildcard* parameter specifies the portion of the source IP host address to match against. 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 to 255, for example 0.0.0.255. Zeros in the mask mean the packet's source address must match the *source-ip* . Ones mean any value matches. For example, the source-ip and wildcard values 10.157.22.26 0.0.0.255 mean that all hosts in the Class C subnet 10.157.22.x match the policy.

If you prefer to specify the wildcard (mask value) in Classless Interdomain Routing (CIDR) format, you can enter a forward slash after the IP address, then enter the number of significant bits in the mask. For example, you can enter the CIDR equivalent of "10.157.22.26 0.0.0.255" as "10.157.22.26/24". The CLI automatically converts the CIDR number into the appropriate ACL mask (where zeros instead of ones are the significant bits) and changes the non-significant portion of the IP address into zeros. For example, if you specify 10.157.22.26/24 or 10.157.22.26 0.0.0.255, then save the changes to the startup-config file, the value appears as 10.157.22.0/24 (if you have enabled display of subnet lengths) or 10.157.22.0 0.0.0.255 in the startup-config file.

If you enable the software to display IP subnet masks in CIDR format, the mask is saved in the file in "/ mask-bits " format. To enable the software to display the CIDR masks, enter the **ip show-subnet-length** command at the global CONFIG level of the CLI. You can use the CIDR format to configure the ACL entry regardless of whether the software is configured to display the masks in CIDR format.

> **NOTE**
> If you use the CIDR format, the ACL entries appear in this format in the running-config and startup-config files, but are shown with subnet mask in the display produced by the **show ip access-list** command.

The *destination-ip* | *hostname* parameter specifies the destination IP host for the policy. If you want the policy to match on all destination addresses, enter **any**.

The *icmp-type* | *icmp-num* parameter specifies the ICMP protocol type:

- This parameter applies only if you specified **icmp** as the ip-protocol value.
- If you use this parameter, the ACL entry is sent to the CPU for processing.
- If you do not specify a message type, the ACL applies to all types of ICMP messages.

The *icmp-num* parameter can be a value from 0 through 255.

The *icmp-type* parameter can have one of the following values, depending on the software version the device is running:

- any-icmp-type
- echo
- echo-reply
- information-request
- log
- mask-reply

- mask-request
- parameter-problem
- redirect
- source-quench
- time-exceeded
- timestamp-reply
- timestamp-request
- traffic policy
- unreachable
- num

NOTE
The QoS options listed below are only available if a specific ICMP type is specified for the icmp-type parameter and cannot be used with the **any-icmp-type** option above. See QoS options for IP ACLs on page 143 for more information on using ACLs to perform QoS.

The tcp/udp comparison operator parameter specifies a comparison operator for the TCP or UDP port number. This parameter applies only when you specify **tcp** or **udp** as the IP protocol. For example, if you are configuring an entry for HTTP, specify **tcp eq http** . You can enter one of the following operators:

- **eq** - The policy applies to the TCP or UDP port name or number you enter after **eq** .
- **established** - This operator applies only to TCP packets. If you use this operator, the policy applies to TCP packets that have the ACK (Acknowledgment) or RST (Reset) bits set on (set to "1") in the Control Bits field of the TCP packet header. Thus, the policy applies only to established TCP sessions, not to new sessions. Refer to Section 3.1, "Header Format", in RFC 793 for information about this field.

  NOTE
  This operator applies only to destination TCP ports, not source TCP ports.

- **gt** - The policy applies to TCP or UDP port numbers greater than the port number or the numeric equivalent of the port name you enter after **gt** .
- **lt** - The policy applies to TCP or UDP port numbers that are less than the port number or the numeric equivalent of the port name you enter after **lt** .
- **neq** - The policy applies to all TCP or UDP port numbers except the port number or port name you enter after **neq** .
- **range** - The policy applies to all TCP or UDP port numbers that are between the first TCP or UDP port name or number and the second one you enter following the **range** parameter. The range includes the port names or numbers you enter. For example, to apply the policy to all ports between and including 23 (Telnet) and 53 (DNS), enter the following: **range 23 53** . The first port number in the range must be lower than the last number in the range.

The *tcp/udp-port* parameter specifies the TCP or UDP port number or well-known name. You can specify a well-known name for any application port whose number is less than 1024. For other application ports, you must enter the number. Enter "?" instead of a port to list the well-known names recognized by the CLI.

The **in** | **out** parameter specifies that the ACL applies to incoming traffic on the interface to which you apply the ACL. You can apply the ACL to an Ethernet port or a virtual interface.

NOTE
If the ACL is for a virtual routing interface, you also can specify a subset of ports within the VLAN containing that interface when assigning an ACL to the interface. Refer to Configuring standard numbered ACLs on page 114.

The **precedence** *name* | *num* parameter specifies the IP precedence. The precedence option for an IP packet is set in a three-bit field following the four-bit header-length field of the packet's header. You can specify one of the following:

- **critical** or **5** - The ACL matches packets that have the critical precedence. If you specify the option number instead of the name, specify number 5.
- **flash** or **3** - The ACL matches packets that have the flash precedence. If you specify the option number instead of the name, specify number 3.
- **flash-override** or **4** - The ACL matches packets that have the flash override precedence. If you specify the option number instead of the name, specify number 4.
- **immediate** or **2** - The ACL matches packets that have the immediate precedence. If you specify the option number instead of the name, specify number 2.
- **internet** or **6** - The ACL matches packets that have the internetwork control precedence. If you specify the option number instead of the name, specify number 6.
- **network** or **7** - The ACL matches packets that have the network control precedence. If you specify the option number instead of the name, specify number 7.
- **priority** or **1** - The ACL matches packets that have the priority precedence. If you specify the option number instead of the name, specify number 1.
- **routine** or **0** - The ACL matches packets that have the routine precedence. If you specify the option number instead of the name, specify number 0.

The **tos** *name* | *num* parameter of the **ip access-list** command specifies the IP ToS. You can specify one of the following:

- **max-reliability** or **2** - The ACL matches packets that have the maximum reliability ToS. The decimal value for this option is 2.
- **max-throughput** or **4** - The ACL matches packets that have the maximum throughput ToS. The decimal value for this option is 4.
- **min-delay** or **8** - The ACL matches packets that have the minimum delay ToS. The decimal value for this option is 8.
- **min-monetary-cost** or **1** - The ACL matches packets that have the minimum monetary cost ToS. The decimal value for this option is 1.

  NOTE
  This value is not supported on 10 Gigabit Ethernet modules.

- – **normal** or **0** - The ACL matches packets that have the normal ToS. The decimal value for this option is 0.
  – *num* - A number from 0 - 15 that is the sum of the numeric values of the options you want. The ToS field is a four-bit field following the Precedence field in the IP header. You can specify one or more of the following. To select more than one option, enter the decimal value that is equivalent to the sum of the numeric values of all the ToS options you want to select. For example, to select the **max-reliability** and **min-delay** options, enter number 10. To select all options, select 15.

  NOTE
  The following QoS options are only available if a specific ICMP type is specified and cannot be used with the **any-icmp-type** option set for the icmp-type parameter. See QoS options for IP ACLs on page 143 for more information on using ACLs to perform QoS.

The **802.1p-priority-matching** option inspects the 802.1p bit in the ACL that can be used with adaptive rate limiting. Enter a value from 0 - 7. For details, refer to "Inspecting the 802.1p bit in the ACL for adaptive rate limiting" section in the *Brocade FastIron Traffic Management Configuration Guide***.**

The **dscp-marking** option enables you to configure an ACL that marks matching packets with a specified DSCP value. Enter a value from 0 - 63.

The **dscp-matching** option matches on the packet's DSCP value. Enter a value from 0 - 63. This option does not change the packet's forwarding priority through the device or mark the packet. Refer to DSCP matching on page 144.

The **log** parameter enables SNMP traps and Syslog messages for inbound packets denied by the ACL:

- You can enable logging on inbound ACLs and filters that support logging even when the ACLs and filters are already in use. To do so, re-enter the **ACL** or **filter** command and add the log parameter to the end of the ACL or filter. The software replaces the **ACL** or **filter** command with the new one. The new ACL or filter, with logging enabled, takes effect immediately.

The **traffic-policy** option enables the device to rate limit inbound traffic and to count the packets and bytes per packet to which ACL permit or deny clauses are applied. For configuration procedures and examples, refer to the chapter "Traffic Policies" in the *Brocade FastIron Traffic Management Configuration Guide*.

To configure an extended named ACL, enter the **ip access-list extended** command.

```
device(config)#ip access-list extended "block Telnet"
device(config-ext-nACL)#deny tcp host 10.157.22.26 any eq telnet log
device(config-ext-nACL)#permit ip any any
device(config-ext-nACL)#exit
device(config)#interface ethernet 1/1/1
device(config-if-1/1/1)#ip access-group "block Telnet" in
```

The options at the ACL configuration level and the syntax for the **ip access-group** command are the same for numbered and named ACLs and are described in Extended numbered ACL configuration on page 118 and Extended numbered ACL configuration on page 118.

Extended named ACL configuration

The commands for configuring named ACL entries are different from the commands for configuring numbered ACL entries. The command to configure a numbered ACL is **access-list**. The command for configuring a named ACL is **ip access-list**. In addition, when you configure a numbered ACL entry, you specify all the command parameters on the same command. When you configure a named ACL, you specify the ACL type (standard or extended) and the ACL number with one command, which places you in the configuration level for that ACL. Once you enter the configuration level for the ACL, the command syntax is the same as the syntax for numbered ACLs.

Extended ACLs let you permit or deny packets based on the following information:

- IP protocol
- Source IP address or host name
- Destination IP address or host name
- Source TCP or UDP port (if the IP protocol is TCP or UDP)
- Destination TCP or UDP port (if the IP protocol is TCP or UDP)

The IP protocol can be one of the following well-known names or any IP protocol number from 0 - 255:

- Internet Control Message Protocol (ICMP)
- Internet Group Management Protocol (IGMP)
- Internet Gateway Routing Protocol (IGRP)
- Internet Protocol (IP)
- Open Shortest Path First (OSPF)

- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

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 IP address to the website's IP address.

Extended named ACL syntax

Syntax: [no] ip access-list extended *ACL-name* { **deny** | **permit** } *ip-protocol* { *source-ip* | *hostname wildcard* } [*operator* [*source-tcp* | *udp-port*]] | *destination-ip* | *hostname* [*icmp-num* | *icmp-type*] *wildcard* [**tcp** | **udp**] *comparison operator destination* [**tcp** | **udp** *port*] [**802.1p-priority-matching** *0-7*] [**dscp-cos-mapping**] [**dscp-marking** *0-63* [**802.1p-priority-marking** *0-7...* | **dscp-cos-mapping**]] [**dscp-matching** *0-63*] [**log**] [**precedence** *name* | *0-7*] [**tos** *0-63* | *name*] [**traffic-policy** *name*]

Syntax: [no] ip access-group *num* [**in** | **out**]

The *ACL-name* parameter is the access list name. You can specify a string of up to 255 alphanumeric characters. You can use blanks in the ACL name if you enclose the name in quotation marks (for example, "ACL for Net1"). ACL names must be unique across all IPv4 and IPv6 ACLs.

The **deny** | **permit** parameter indicates whether packets that match the policy are dropped or forwarded.

The *ip-protocol* parameter indicates the type of IP packet you are filtering. You can specify a well-known name for any protocol whose number is less than 255. For other protocols, you must enter the number. Enter "?" instead of a protocol to list the well-known names recognized by the CLI.

The *source-ip* | *hostname* parameter specifies the source IP host for the policy. If you want the policy to match on all source addresses, enter **any** .

The *wildcard* parameter specifies the portion of the source IP host address to match against. 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 to 255, for example 0.0.0.255. Zeros in the mask mean the packet's source address must match the *source-ip* . Ones mean any value matches. For example, the source-ip and wildcard values 10.157.22.26 0.0.0.255 mean that all hosts in the Class C subnet 10.157.22.x match the policy.

If you prefer to specify the wildcard (mask value) in Classless Interdomain Routing (CIDR) format, you can enter a forward slash after the IP address, then enter the number of significant bits in the mask. For example, you can enter the CIDR equivalent of "10.157.22.26 0.0.0.255" as "10.157.22.26/24". The CLI automatically converts the CIDR number into the appropriate ACL mask (where zeros instead of ones are the significant bits) and changes the non-significant portion of the IP address into zeros. For example, if you specify 10.157.22.26/24 or 10.157.22.26 0.0.0.255, then save the changes to the startup-config file, the value appears as 10.157.22.0/24 (if you have enabled display of subnet lengths) or 10.157.22.0 0.0.0.255 in the startup-config file.

If you enable the software to display IP subnet masks in CIDR format, the mask is saved in the file in "/ mask-bits " format. To enable the software to display the CIDR masks, enter the **ip show-subnet-length** command at the global CONFIG level of the CLI. You can use the CIDR format to configure the ACL entry regardless of whether the software is configured to display the masks in CIDR format.

> **NOTE**
> If you use the CIDR format, the ACL entries appear in this format in the running-config and startup-config files, but are shown with subnet mask in the display produced by the **show ip access-list** command.

The *destination-ip* | *hostname* parameter specifies the destination IP host for the policy. If you want the policy to match on all destination addresses, enter **any**.

The *icmp-type* | *icmp-num* parameter specifies the ICMP protocol type:

- This parameter applies only if you specified **icmp** as the ip-protocol value.
- If you use this parameter, the ACL entry is sent to the CPU for processing.
- If you do not specify a message type, the ACL applies to all types of ICMP messages.

The *icmp-num* parameter can be a value from 0 through 255.

The *icmp-type* parameter can have one of the following values, depending on the software version the device is running:

- any-icmp-type
- echo
- echo-reply
- information-request
- log
- mask-reply
- mask-request
- parameter-problem
- redirect
- source-quench
- time-exceeded
- timestamp-reply
- timestamp-request
- traffic policy
- unreachable
- num

  NOTE
  The QoS options listed below are only available if a specific ICMP type is specified for the icmp-type parameter and cannot be used with the **any-icmp-type** option above. See QoS options for IP ACLs on page 143 for more information on using ACLs to perform QoS.

The tcp/udp comparison operator parameter specifies a comparison operator for the TCP or UDP port number. This parameter applies only when you specify **tcp** or **udp** as the IP protocol. For example, if you are configuring an entry for HTTP, specify **tcp eq http** . You can enter one of the following operators:

- **eq** - The policy applies to the TCP or UDP port name or number you enter after **eq** .
- **established** - This operator applies only to TCP packets. If you use this operator, the policy applies to TCP packets that have the ACK (Acknowledgment) or RST (Reset) bits set on (set to "1") in the Control Bits field of the TCP packet header. Thus, the policy applies only to established TCP sessions, not to new sessions. Refer to Section 3.1, "Header Format", in RFC 793 for information about this field.

  NOTE
  This operator applies only to destination TCP ports, not source TCP ports.

- **gt** - The policy applies to TCP or UDP port numbers greater than the port number or the numeric equivalent of the port name you enter after **gt** .
- **lt** - The policy applies to TCP or UDP port numbers that are less than the port number or the numeric equivalent of the port name you enter after **lt** .

- **neq** - The policy applies to all TCP or UDP port numbers except the port number or port name you enter after **neq** .
- **range** - The policy applies to all TCP or UDP port numbers that are between the first TCP or UDP port name or number and the second one you enter following the **range** parameter. The range includes the port names or numbers you enter. For example, to apply the policy to all ports between and including 23 (Telnet) and 53 (DNS), enter the following: **range 23 53** . The first port number in the range must be lower than the last number in the range.

The *tcp/udp-port* parameter specifies the TCP or UDP port number or well-known name. You can specify a well-known name for any application port whose number is less than 1024. For other application ports, you must enter the number. Enter "?" instead of a port to list the well-known names recognized by the CLI.

The **in** | **out** parameter specifies that the ACL applies to incoming traffic on the interface to which you apply the ACL. You can apply the ACL to an Ethernet port or a virtual interface.

NOTE
If the ACL is for a virtual routing interface, you also can specify a subset of ports within the VLAN containing that interface when assigning an ACL to the interface. Refer to Configuring standard numbered ACLs on page 114.

The **precedence** *name* | *num* parameter specifies the IP precedence. The precedence option for an IP packet is set in a three-bit field following the four-bit header-length field of the packet's header. You can specify one of the following:

- **critical** or **5** - The ACL matches packets that have the critical precedence. If you specify the option number instead of the name, specify number 5.
- **flash** or **3** - The ACL matches packets that have the flash precedence. If you specify the option number instead of the name, specify number 3.
- **flash-override** or **4** - The ACL matches packets that have the flash override precedence. If you specify the option number instead of the name, specify number 4.
- **immediate** or **2** - The ACL matches packets that have the immediate precedence. If you specify the option number instead of the name, specify number 2.
- **internet** or **6** - The ACL matches packets that have the internetwork control precedence. If you specify the option number instead of the name, specify number 6.
- **network** or **7** - The ACL matches packets that have the network control precedence. If you specify the option number instead of the name, specify number 7.
- **priority** or **1** - The ACL matches packets that have the priority precedence. If you specify the option number instead of the name, specify number 1.
- **routine** or **0** - The ACL matches packets that have the routine precedence. If you specify the option number instead of the name, specify number 0.

The **tos** *name* | *num* parameter of the **ip access-list** command specifies the IP ToS. You can specify one of the following:

- **max-reliability** or **2** - The ACL matches packets that have the maximum reliability ToS. The decimal value for this option is 2.
- **max-throughput** or **4** - The ACL matches packets that have the maximum throughput ToS. The decimal value for this option is 4.
- **min-delay** or **8** - The ACL matches packets that have the minimum delay ToS. The decimal value for this option is 8.
- **min-monetary-cost** or **1** - The ACL matches packets that have the minimum monetary cost ToS. The decimal value for this option is 1.

  NOTE
  This value is not supported on 10 Gigabit Ethernet modules.

- – **normal** or **0** - The ACL matches packets that have the normal ToS. The decimal value for this option is 0.

- *num* - A number from 0 - 15 that is the sum of the numeric values of the options you want. The ToS field is a four-bit field following the Precedence field in the IP header. You can specify one or more of the following. To select more than one option, enter the decimal value that is equivalent to the sum of the numeric values of all the ToS options you want to select. For example, to select the **max-reliability** and **min-delay** options, enter number 10. To select all options, select 15.

NOTE
The following QoS options are only available if a specific ICMP type is specified and cannot be used with the **any-icmp-type** option set for the icmp-type parameter. See QoS options for IP ACLs on page 143 for more information on using ACLs to perform QoS.

The **802.1p-priority-matching** option inspects the 802.1p bit in the ACL that can be used with adaptive rate limiting. Enter a value from 0 - 7. For details, refer to "Inspecting the 802.1p bit in the ACL for adaptive rate limiting" section in the *Brocade FastIron Traffic Management Configuration Guide***.**

The **dscp-marking** option enables you to configure an ACL that marks matching packets with a specified DSCP value. Enter a value from 0 - 63.

The **dscp-matching** option matches on the packet's DSCP value. Enter a value from 0 - 63. This option does not change the packet's forwarding priority through the device or mark the packet. Refer to DSCP matching on page 144.

The **log** parameter enables SNMP traps and Syslog messages for inbound packets denied by the ACL:

- You can enable logging on inbound ACLs and filters that support logging even when the ACLs and filters are already in use. To do so, re-enter the **ACL** or **filter** command and add the log parameter to the end of the ACL or filter. The software replaces the **ACL** or **filter** command with the new one. The new ACL or filter, with logging enabled, takes effect immediately.

The **traffic-policy** option enables the device to rate limit inbound traffic and to count the packets and bytes per packet to which ACL permit or deny clauses are applied. For configuration procedures and examples, refer to the chapter "Traffic Policies" in the *Brocade FastIron Traffic Management Configuration Guide*.

To configure an extended named ACL, enter the **ip access-list extended** command.

```
device(config)#ip access-list extended "block Telnet"
device(config-ext-nACL)#deny tcp host 10.157.22.26 any eq telnet log
device(config-ext-nACL)#permit ip any any
device(config-ext-nACL)#exit
device(config)#interface ethernet 1/1/1
device(config-if-1/1/1)#ip access-group "block Telnet" in
```

The options at the ACL configuration level and the syntax for the **ip access-group** command are the same for numbered and named ACLs and are described in Extended numbered ACL configuration on page 118 and Extended numbered ACL configuration on page 118.

Applying egress ACLs to Control (CPU) traffic

By default, outbound ACLs are not applied to traffic generated by the CPU. This must be enabled using the **enable egress-acl-on-control-traffic** command.

Syntax: enable egress-acl-on-control-traffic

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
```

Syntax: 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-list 140** reverts to the port name for the TCP port (http in this example). After the user issues the new **ip preserve-ACL-user-input-format** command, **show ip access-list 140** displays either the TCP port number or name, depending on how it was configured by the user.

```
device(config)#access-list 140 permit tcp any any eq 80
device(config)#access-list 140 permit tcp any any eq ftp
device#show ip access-lists 140
Extended IP access list 140
permit tcp any any eq http
permit tcp any any eq ftp
device(config)#access-list 140 permit tcp any any eq 80
device(config)#access-list 140 permit tcp any any eq ftp
device#show ip access-lists 140
Extended IP access list 140
permit tcp any any eq http
permit tcp any any eq ftp
device(config)#ip preserve-ACL-user-input-format
device#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. The comment text appears in the output of **show** commands that display ACL information.

This section describes how to add, delete, and view ACL comments.

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)#access-list 100 remark The following line permits TCP packets
device(config)#access-list 100 permit tcp 192.168.4.40/24 2.2.2.2/24
device(config)#access-list 100 remark The following permits UDP packets
device(config)#access-list 100 permit udp 192.168.2.52/24 2.2.2.2/24
device(config)#access-list 100 deny ip any any
```

You can add comments to entries in a numbered ACL using the syntax for named ACLs. For example, using the same example configuration above, you could instead enter the following commands.

```
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
```

Syntax: [no] access-list *ACL-num* **remark** *comment-text*

or

Syntax: [no] ip access-list [**standard** | **extended**] *ACL-num*

Syntax:remark *comment-text*

For *ACL-num* , enter the number of the ACL.

The *comment-text* can be up to 128 characters in length. The comment must be entered separately from the actual ACL entry; that is, you cannot enter the ACL entry and the ACL comment with the same **access-list** or **ip access-list** command. Also, in order for the remark to be displayed correctly in the output of **show** commands, the comment must be entered immediately before the ACL entry it describes. Note that an ACL comment is tied to the ACL entry immediately following the comment. Therefore, if the ACL entry is removed, the ACL comment is also removed.

The **standard** | **extended** parameter indicates the ACL type.

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
```

Syntax: [no] access-list [**standard** | **extended**] *ACL-name* **remark** *comment-text*

The **standard** | **extended** parameter indicates the ACL type.

For *ACL-name*, enter the name of the ACL.

The *comment-text* can be up to 128 characters in length. The comment must be entered separately from the actual ACL entry; that is, you cannot enter the ACL entry and the ACL comment with the same **ip access-list** command. Also, in order for the remark to be displayed correctly in the output of show commands, the comment must be entered immediately before the ACL entry it describes. Note that an ACL comment is tied to the ACL entry immediately following the comment. Therefore, if the ACL entry is removed, the ACL comment is also removed.

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)#no remark The following line permits TCP packets
```

Syntax: [no] remark *comment-text*

Viewing comments in an ACL

You can use the following commands to display comments for IPv4 ACLs:

- **show running-config**
- **show access-list**
- **show access-list named-acl**
- **show ip access-list**

NOTE
For details of these commands, refer to the *FastIron Command Reference*.

The following shows the comment text for a numbered ACL, ACL 100, in a **show running-config** display.

```
device#show running-config
...
access-list 100 remark The following line permits TCP packets
access-list 100 permit tcp 192.168.4.40/24 2.2.2.2/24
access-list 100 remark The following line permits UDP packets
access-list 100 permit udp 192.168.2.52/24 2.2.2.2/24
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 ethe 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 ethe 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)#int e 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)#int 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#
```

ACL logging

Brocade devices support ACL logging of inbound packets that are sent to the CPU for processing (denied packets).

NOTE
ACL logging is not supported for outbound packets or any packets that are processed in hardware (permitted packets).

You may want the software to log entries in the Syslog for packets that are denied by ACL filters. ACL logging is disabled by default; it must be explicitly enabled on a port.

When you enable logging for ACL entries, statistics for packets that match the deny conditions of the ACL entries are logged. For example, if you configure a standard ACL entry to deny all packets from source address 10.157.22.26, statistics for packets that are explicitly denied by the ACL entry are logged in the Syslog buffer and in SNMP traps sent by the Brocade device.

The first time an ACL entry denies a packet, the software immediately generates a Syslog entry and an SNMP trap. The software also starts a five-minute timer. The timer keeps track of all packets explicitly denied by the ACL entries. After five minutes, the software generates a single Syslog entry for each ACL entry that denied a packet. The Syslog entry (message) indicates the number of packets denied by the ACL entry during the previous five minutes. Note however, that packet count may be inaccurate if the packet rate is high and exceeds the CPU processing rate.

If no ACL entries explicitly deny packets during an entire five-minute timer interval, the timer stops. The timer restarts when an ACL entry explicitly denies a packet.

NOTE
The timer for logging packets denied by MAC address filters is a different timer than the ACL logging timer.

Configuration notes for ACL logging

Note the following points before configuring ACL logging:

- ACL logging is supported for denied packets, which are sent to the CPU for logging. ACL logging is not supported for permitted packets.
- ACL logging is not supported for dynamic ACLs with MAC authentication or 802.1X enabled.
- Packets that are denied by ACL filters are logged in the Syslog based on a sample time-period.
- You can enable ACL logging on physical and virtual interfaces.
- When ACL logging is disabled, packets that match the ACL rule are forwarded or dropped in hardware.
- ACL logging is supported for ACLs that are applied to network management access features such as Telnet, SSH, and SNMP.
- When an ACL that includes an entry with a logging option is applied 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.

NOTE
The above limitation applies only to IPv4 ACLs, it does not apply to the use of ACLs to log IPv6 traffic.

- When ACL logging is enabled, packets sent to the CPU are automatically rate-limited to prevent CPU overload.
- ACL logging is intended for debugging purposes. Ruckus recommends that you disable ACL logging after the debug session is over.

Configuration tasks for ACL logging

To enable ACL logging, complete the following steps:

1. Create ACL entries with the log option
2. Enable ACL logging on individual ports

   NOTE
   The command syntax for enabling ACL logging is different on IPv4 devices than on IPv6 devices. See the configuration examples in the next section.

3. Bind the ACLs to the ports on which ACL logging is enabled

Example ACL logging configuration

The following shows an example ACL logging configuration on an IPv4 device.

```
device(config)#access-list 1 deny host 10.157.22.26 log
device(config)#access-list 1 deny 10.157.29.12 log
device(config)#access-list 1 deny host IPHost1 log
device(config)#access-list 1 permit any
device(config)#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 above commands create ACL entries that include the log option, enable ACL logging on interface e 1/1/4, then bind the ACL to interface e 1/1/4. Statistics for packets that match the deny statements will be logged.

Syntax: ACL-logging

The **ACL-logging** command applies to IPv4 devices only. For IPv6 devices, use the **logging-enable** command as shown in the following example.

The following shows an example configuration on an IPv6 device.

```
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
```

The above commands create ACL entries that include the log option, then bind the ACL to interface ethernet 1/9/12. Statistics for packets that match the deny statement will be logged.

Syntax: logging-enable

NOTE
The **logging-enabled** command applies to IPv6 devices only. For IPv4 devices, use the **ACL-logging** command as shown in the previous example.

Displaying ACL Log Entries

The first time an entry in an ACL permits or denies a packet and logging is enabled for that entry, the software generates a Syslog message and an SNMP trap. Messages for packets permitted or denied by ACLs are at the warning level of the Syslog.

When the first Syslog entry for a packet permitted or denied by an ACL is generated, the software starts an ACL timer. After this, the software sends Syslog messages every five minutes. If an ACL entry does not permit or deny any packets during the timer interval, the software does not generate a Syslog entry for that ACL entry.

NOTE
For an ACL entry to be eligible to generate a Syslog entry for denied packets, logging must be enabled for the entry. The Syslog contains entries only for the ACL entries that deny packets and have logging enabled.

To display Syslog entries, enter the **show log** command from any CLI prompt:

```
device#show log
Syslog logging: enabled (0 messages dropped, 2 flushes, 0 overruns)
 Buffer logging: level ACDMEINW, 9 messages logged
 level code: A=alert C=critical D=debugging M=emergency E=error
  I=informational N=notification W=warning
Dynamic Log Buffer (50 lines):
0d00h12m18s:W:ACL: ACL: List 122 denied tcp 10.20.15.6(0)(Ethernet 4
0000.0004.01) -> 10.20.18.6(0), 1 event(s)
0d00h12m18s:W:ACL: ACL: List 122 denied tcp 10.20.15.2(0)(Ethernet 4
0000.0004.01) -> 10.20.18.2(0), 1 event(s)
0d00h12m18s:W:ACL: ACL: List 122 denied tcp 10.20.15.4(0)(Ethernet 4
0000.0004.01) -> 10.20.18.4(0), 1 event(s)
0d00h12m18s:W:ACL: ACL: List 122 denied tcp 10.20.15.3(0)(Ethernet 4
0000.0004.01) -> 10.20.18.3(0), 1 event(s)
0d00h12m18s:W:ACL: ACL: List 122 denied tcp 10.20.15.5(0)(Ethernet 4
0000.0004.01) -> 10.20.18.5(0), 1 event(s)
0d00h12m18s:I:ACL: 122 applied to port 4 by from console session
0d00h10m12s:I:ACL: 122 removed from port 4 by from console session
0d00h09m56s:I:ACL: 122 removed from port 4 by from console session
0d00h09m38s:I:ACL: 122 removed from port 4 by from console session
```

Syntax: show log

Enabling strict control of ACL filtering of fragmented packets

The default processing of fragments by hardware-based ACLs is as follows:

- 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, since 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.
- For other fragments of the same packet, they 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, since a transaction cannot be completed without the entire packet.

For tighter control, 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
Brocade(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.

Syntax: [no] ip access-group frag deny

ACL support for switched traffic in the router image

ACL support for switched traffic—inbound and outbound—in the router image is enabled by default.

Enabling ACL filtering based on VLAN membership or VE port membership

> **NOTE**
> This section applies to IPv4 ACLs only. IPv6 ACLs do not support ACL filtering based on VLAN membership or VE port membership. This feature is not applicable to outbound traffic.

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). By default, this feature support is disabled. To enable it, enter the following commands at the Global CONFIG level of the CLI.

```
device(config)#enable ACL-per-port-per-vlan
device(config)#write memory
device(config)#exit
device#reload
```

> **NOTE**
> For complete configuration examples, see Applying an IPv4 ACL to specific VLAN members on a port (Layer 2 devices only) on page 139 and Applying an IPv4 ACL to a subset of ports on a virtual interface (Layer 3 devices only) on page 139.

Syntax: [no] enable ACL-per-port-per-vlan *VLAN-ID*

Enter the **no** form of the command to disable this feature.

Configuration notes under acls-per-port-per-vlan

- Brocade 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 (ACL-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) You cannot change VLAN membership on a port while **acl-per-port-per-vlan** is enabled.

Applying an IPv4 ACL to specific VLAN members on a port (Layer 2 devices only)

NOTE
This section applies to IPv4 ACLs only. IPv6 ACLs do not support ACL filtering based on VLAN membership.

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 packets' VLAN membership.

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)#untag ethernet 5 to 8
device(config-vlan-12)#tag ethernet 23 to 24
device(config-vlan-12)#exit
device(config)#access-list 10 deny host 10.157.22.26 log
device(config)#access-list 10 deny 10.157.29.12 log
device(config)#access-list 10 deny host IPHost1 log
device(config)#access-list 10 permit
device(config)#int e 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 this example configure port-based VLAN 12, and add ports e 5 - 8 as untagged ports and ports e 23 - 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 port e 23 is a member.

Syntax: [no] enable ACL-per-port-per-vlan *VLAN-ID*

Syntax: [no] ip access-group *ACL-ID*

The *VLAN ID* parameter specifies the VLAN name or number to which you will bind the ACL.

The *ACL ID* parameter is the access list name or number.

Applying an IPv4 ACL to a subset of ports on a virtual interface (Layer 3 devices only)

NOTE
This section applies to IPv4 ACLs only. IPv6 ACLs do not support ACL filtering based on VE port membership.

You can apply an IPv4 ACL to a virtual routing interface. The virtual interface is used for routing between VLANs and contains all the ports within the VLAN. The IPv4 ACL applies to all the ports on the virtual routing interface. You also can specify a subset of ports within the VLAN containing a specified virtual interface when assigning an ACL to that virtual interface.

Use this feature 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)#access-list 1 deny host 10.157.22.26 log
device(config)#access-list 1 deny 10.157.29.12 log
device(config)#access-list 1 deny host IPHost1 log
device(config)#access-list 1 permit any
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 this example configure port-based VLAN 10, add ports 1/1/1 - 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.

Syntax: [no] ip access-group *ACL-ID* **in** *interface port* [**to** *port*]

The *ACL ID* parameter is the access list name or number.

ACLs to filter ARP packets

NOTE
This feature is not applicable to outbound traffic.

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. Since 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 to be written in the ARP table; others are dropped.

Configuration considerations for filtering ARP packets

- This feature is available on devices running Layer 3 code. This filtering occurs on the management processor.
- The feature 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 ACLs which are with protocol IP only can be used. If any other ACL is used, an error is displayed.

Configuring ACLs for ARP filtering

To implement the ACL ARP filtering feature, enter commands such as the following.

```
device(config)# access-list 101 permit ip host 192.168.2.2 any
device(config)# access-list 102 permit ip host 192.168.2.3 any
device(config)# access-list 103 permit ip host 192.168.2.4 any
device(config)# vlan 2
device(config-vlan-2)# tag ethe 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)# tag ethe 1/1/1 to 1/1/2
device(config-vlan-3)#router-int ve 3
device(config-vlan-3)# vlan 4
device(config-vlan-4)# tag ethe 1/1/1 to 1/1/2
device(config-vlan-4)# router-int 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
```

Syntax: [no] ip use-ACL-on-arp [*access-list-number*]

When the **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 to be written in the ARP table; those that are not permitted will be dropped.

The access-list-number parameter 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. You can do one of the following for access-list-number :

- Enter an ACL ID to explicitly specify the ACL to be used for filtering. In the example above, the line device`#ip use-ACL-on-arp` 103 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 above, the line device`#ip use-ACL-on-arp` 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 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 a command such as the following.

```
device(config)#show ACL-on-arp
Port ACL ID Filter Count
2    103    10
3    102    23
4    101    12
```

Syntax: show ACL-on-arp [*interface port*] | **loopback** [*num*] | **ve** [*num*]]

If the *port* variable is not specified, all ports on the device that use ACLs for ARP filtering will be included in the display.

The Filter Count column 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 a command such as the following.

```
device(config)#clear ACL-on-arp
```

The above command resets the filter count on all interfaces in a device back to zero.

Syntax: clear ACL-on-arp

Filtering on IP precedence and ToS values

To configure an extended IP ACL that matches based on IP precedence, enter commands such as the following.

```
device(config)#access-list 103 deny tcp 10.157.21.0/24 10.157.22.0/24
precedence internet
device(config)#access-list 103 deny tcp 10.157.21.0/24 eq ftp 10.157.22.0/24
precedence 6
device(config)#access-list 103 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.

To configure an IP ACL that matches based on ToS, enter commands such as the following.

```
device(config)#access-list 104 deny tcp 10.157.21.0/24 10.157.22.0/24 tos
normal
device(config)#access-list 104 deny tcp 10.157.21.0/24 eq ftp 10.157.22.0/24
tos 13
device(config)#access-list 104 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.

TCP flags - edge port security

The edge port security feature works in combination with IP ACL rules and can be combined with other ACL functions (such as dscp-marking and traffic policies), giving you greater flexibility when designing ACLs.

For details about the edge port security feature, refer to the *Using TCP Flags in combination with other ACL features* section.

QoS options for IP ACLs

Quality of Service (QoS) options 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 (described below). This option maps the DSCP value in incoming packets to a hardware table that provides mapping of each of the 0 - 63 DSCP values, and distributes them among eight traffic classes (internal priorities) and eight 802.1p priorities.

By default, the Ruckus device does the *802.1p* to *CoS* mapping. If you want to change the priority mapping to *DSCP* to *CoS* mapping, you must enter the following ACL statement.

```
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 re-mark 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
  These QoS options are only available if a specific ICMP type is specified for the icmp-type parameter while configuring extended ACLS, and cannot be used with the **any-icmp-type** option. See Extended numbered ACL syntax on page 119 and Extended named ACL configuration on page 127for the syntax for configuring extended ACLs.

Configuration notes for QoS options

- These devices do not support marking and prioritization simultaneously with the same rule (and do not support DSCP CoS mapping at all). To achieve this, you need to 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)#access-list 101 permit ip any any dscp-marking 43
```

or

```
device(config)#access-list 101 permit ip any any 802.1p-priority-marking
```

or

```
device(config)#access-list 101 permit ip any any internal-priority-marking 6
```

or

```
device(config)#access-list 101 permit ip any any dscp-marking 43
802.1p-priority-marking 4 internal-priority-marking 6
```

Using a combined ACL for 802.1p marking

Ruckus devices support a simple method for assigning an 802.1p priority value to packets without affecting the actual packet or the DSCP. In early IronWare software releases, users were required to provide DSCP-marking and DSCP-matching information in order to assign 802.1p priority values, which required the deployment of a 64-line ACL to match all possible DSCP values. Users were also required to configure an internal priority marking value. Now, users can easily specify 802.1p priority marking values directly, and change internal priority marking from *required* to o*ptional* .

NOTE
This feature is not applicable to outbound traffic.

For IP

```
device(config)#acc 104 per ip any any 802.1p-priority-marking 1
```

or the following command, which also assigns an optional internal-priority-marking value.

```
device(config)#acc 104 per ip any any 802.1p-priority-marking 1 internal-priority-marking 5
```

Syntax: access-list num *(100-199)* **permit ip any any 802.1p-priority-marking** *priority value 0-7* [**internal-priority-marking** *value 0-7*]

For TCP

```
device(config)#acc 105 per tcp any any 802.1p-priority-marking 1
```

or the following command, which also assigns an optional internal-priority-marking value.

```
device(config)#acc 105 per tcp any any 802.1p-priority-marking 1 internal-priority-marking 5
```

Syntax: access-list num *(100-199)* **permit tcp any any 802.1p-priority-marking** *priority value (0-7)* [**internal-priority-marking** *value (0-7)*]

For UDP

```
device(config)#acc 105 per udp any any 802.1p-priority-marking 1
```

or the following command, which also assigns an optional internal-priority-marking value.

```
device(config)#acc 105 per udp any any 802.1p-priority-marking 1 internal-priority-marking 5
```

Syntax: access-list num *(100-199)* **permit udp any any 802.1p-priority-marking** *priority value (0-7)* [**internal-priority-marking** *value (0-7)*]

In each of these examples, in the first command the internal-priority value is not specified, which means it maintains a default value of 1 (equal to that of the 802.1p value). In the second command, the internal-priority value has been configured by the user to 5.

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)#access-list 112 permit ip 1 0.1.1.0 0.0.0.255 10.2.2.x 0.0.0.255 dscp-matching 29
```

The complete CLI syntax for this feature is shown in Extended numbered ACL configuration on page 118 and Extended named ACL configuration on page 127. The following shows the syntax specific to this feature.

Syntax: ...dscp-matching *0-63*

NOTE
For complete syntax information, refer to Extended numbered ACL syntax on page 119.

ACL-based rate limiting

ACL-based rate limiting provides the facility to limit the rate for IP traffic that matches the permit conditions in extended IP ACLs. This feature is available in the Layer 2 and Layer 3 code.

NOTE
Brocade devices support ACL-based rate limiting for inbound traffic. This feature is not supported for outbound traffic.

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 chapter "Traffic Policies" in the *Brocade FastIron Traffic Management Configuration Guide* .

NOTE
The terms *ACL statistics* and *ACL counting* are used interchangeably in this guide and mean the same thing.

ACL accounting

ACL accounting helps to collect usage information for access 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. Accounting on IPv4 ACLs, IPv6 ACLs, and MAC filters are explained in the corresponding sections of this guide.

Feature limitations for ACL accounting

- Traffic Policer and ACL accounting cannot coexist.
- ACL accounting is not supported on outbound ACLs.
- ACL accounting is not supported on dynamic ACLs.

Configuring IPv4 ACL accounting

Steps to enable, display, and clear IPv4 ACL accounting

On enabling IPv4 ACL accounting for FastIron devices, it will be enabled on all the filters of the ACL including the implicit rule. You can enable ACL accounting for named and numbered ACLs.

1. To enable ACL accounting for a configured ACL, choose one of the following options.
   - For a numbered ACL, use the **access-list enable-accounting** command in the global configuration mode.
   - For a named ACL, use the **enable-accounting** command in the ACL configuration mode.

   ```
   device(config)#access-list 10 enable-accounting

   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 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 how to enable ACL accounting for a numbered ACL.

```
device(config)# access-list 10 permit host 10.10.10.1
device(config)# access-list 10 enable-accounting
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 std
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)# 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 sends advertisement messages to bootstrap routers
- Identify which multicast group packets will be forwarded or blocked on an interface

Enabling and viewing hardware usage statistics for an ACL

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** *access-list-id* command to determine the hardware usage for an ACL. To gain more hardware resources, you can modify the ACL rules so that it uses less hardware resource.

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.

Syntax: show access-list hw-usage [**on** | **off**]

Syntax: show access-list [*access-list-id* | **all**]

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.

The *access-list-id* variable is a valid ACL name or number.

Displaying ACL information

For details of commands for display of IPv4 ACLs, refer to the following *FastIron Command Reference* topics:

- **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 148.
- To determine whether the issue is specific to fragmentation, remove the Layer 4 information (TCP or UDP application ports) from the ACL, 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

IPv6 ACL overview

Brocade devices support IPv6 Access Control Lists (ACLs) for inbound and outbound traffic filtering, as detailed in the *Supported IPv6 ACL features* table. You can configure up to 100 IPv6 ACLs and, by default, up to a system-wide maximum of 4000 ACL rules. For example, you can configure one ACL with 4000 entries, two ACLs with 2000 and 2093 entries respectively (combining IPv4 and IPv6 ACLs), etc.

An IPv6 ACL is composed of one or more conditional statements that pose an action (permit or deny) if a packet matches a specified source or destination prefix. There can be up to 1536 statements per port region, including IPv6, IPv4, MAC address filters, and default statements. ICX 7750 devices have 2048 TCAM rules per-port region. When the maximum number of ACL rules allowed per port region is reached, an error message will display on the console.

In ACLs with multiple statements, you can specify a priority for each statement.The specified priority determines the order in which the statement appears in the ACL. The last statement in each IPv6 ACL is an implicit deny statement for all packets that do not match the previous statements in the ACL.

You can configure an IPv6 ACL on a global basis, then apply it to the incoming or outgoing IPv6 packets on specified interfaces. You can apply only one incoming and only one outgoing IPv6 ACL to an interface. When an interface sends or receives an IPv6 packet, it applies the statements within the ACL in their order of appearance to the packet. As soon as a match occurs, the Brocade device takes the specified action (permit or deny the packet) and stops further comparison for that packet.

IPv6 ACLs are supported on:

- Gbps Ethernet ports
- 10 Gbps Ethernet ports
- Trunk groups
- Virtual routing interfaces

NOTE
IPv6 ACLs are supported on inbound and outbound traffic and are implemented in hardware, making it possible for the Brocade device to filter traffic at line-rate speed on 10 Gbps interfaces.

IPv6 ACL traffic filtering criteria

The Ruckus implementation of IPv6 ACLs enable 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. Hence, 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 - 255:

- Authentication Header (AHP)
- 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.

IPv6 ACL configuration notes

- 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 interface or an IPv6 address should be configured on the interface before an ACL can be applied to it. If IPv6 is not enabled or if there is no IPv6 address configured on the interface, the system will display the following error message.
- On interfaces that have IPv6 ACLs applied on outbound packets, the following features are not supported:
  - ACL mirroring
  - ACL accounting
  - ACL logging
  - Traffic policies
  - Internal priority marking

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 section "IPv6 configuration on each router interface" in the *Brocade FastIron Layer 3 Routing Configuration Guide* and further discussed in Enabling IPv6 on an interface to which an ACL will be applied on page 159.

```
device(config-if-e1000-7)#ipv6 traffic-filter netw in Error: IPv6 is not enabled for interface 7
```

- You cannot disable IPv6 on an interface to which an ACL is bound. Attempting to do so will cause the system to return the following error message.

```
device(config-if-e1000-7)#no ipv6 enable
Error: Port 7 has IPv6 ACL configured. Cannot disable IPv6
```

To disable IPv6, first remove the ACL from the interface.

- For notes on applying IPv6 ACLs to trunk ports, see Applying an IPv6 ACL to a trunk group on page 160.
- For notes on applying IPv6 ACLs to virtual ports, see Applying an IPv6 ACL to a virtual interface in a protocol-based or subnet-based VLAN on page 160.

Configuring an IPv6 ACL

Follow the steps given below to configure an IPv6 ACL.

1. Create the ACL.
2. Enable IPv6 on the interface to which the ACL will be applied.
3. Apply the ACL to the interface.

Example IPv6 configurations

To configure an access list that blocks all Telnet traffic received on port 1/1/1 from IPv6 host 2001:DB8:e0bb::2, enter the following commands.

```
device(config)# ipv6 access-list fdry
device(config-ipv6-access-list-fdry)# deny tcp host 2001:DB8:e0bb::2 any eq
telnet
device(config-ipv6-access-list-fdry)# permit ipv6 any any
device(config-ipv6-access-list-fdry)# exit
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# ipv6 enable
device(config-if-1/1/1)# ipv6 traffic-filter fdry in
device(config)# write memory
```

The following is another example of commands for configuring an ACL and applying it to an interface.

```
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 ipv6 host 2001:DB8:e0ac::2 host
2001:DB8:e0aa:0::24
device(config-ipv6-access-list-netw)# deny udp any any
device(config-ipv6-access-list-netw)# permit ipv6 any any
```

The first condition permits ICMP traffic from hosts in the 2001:DB8:e0bb::x network to hosts in the 2001:DB8::x network.

The second condition denies all IPv6 traffic from host 2001:DB8:e0ac::2 to host 2001:DB8:e0aa:0::24.

The third condition denies all UDP traffic.

The fourth condition permits all packets that are not explicitly denied by the other entries. Without this entry, the ACL would deny all incoming IPv6 traffic on the ports to which you assigned the ACL.

The following commands apply the ACL "netw" to the incoming traffic on port 1/1/2 and to the incoming traffic on port 1/4/3.

```
device(config)# interface ethernet 1/1/2
device(config-if-1/1/2)# ipv6 enable
device(config-if-1/1/2)# ipv6 traffic-filter netw in
device(config-if-1/1/2)# exit
device(config)# interface ethernet 1/4/3
device(config-if-1/4/3)# ipv6 enable
device(config-if-1/4/3)# ipv6 traffic-filter netw in
device(config)# write memory
```

Here is another example.

```
device(config)# ipv6 access-list nextone
device(config-ipv6-access-list rtr)# deny tcp 2001:DB8:21::/24
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)# permit ipv6 any any
device(config-ipv6-access-list rtr)# write memory
```

The first condition in this ACL denies TCP traffic from the 2001:DB8:21::x network to the 2001:DB8:22::x network.

The next condition denies UDP packets from any source with source UDP port in ranges 5 to 6 and whose destination is to the 2001:DB8:22::/24 network.

The third condition permits all packets containing source and destination addresses that are not explicitly denied by the first two. Without this entry, the ACL would deny all incoming IPv6 traffic on the ports to which you assign the ACL.

A **show running-config** command displays the following.

```
device(config)# 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
```

A **show ipv6 access-list** command displays the following.

```
device(config)# sh 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
```

The following commands apply the ACL "rtr" to the incoming traffic on ports 1/2/1 and 1/2/2.

```
device(config)# interface ethernet 1/2/1
device(config-if-1/2/1)# ipv6 enable
device(config-if-1/2/1)# ipv6 traffic-filter rtr in
device(config-if-1/2/1)# exit
device(config)# int eth 1/2/2
device(config-if-1/2/2)# ipv6 enable
```

```
device(config-if-1/2/2)# ipv6 traffic-filter rtr in
device(config)# write memory
```

Default and implicit IPv6 ACL action

The default action when no IPv6 ACLs are configured on an interface is to permit all IPv6 traffic. However, once you configure an IPv6 ACL and apply it to an interface, the default action for that interface is to deny all IPv6 traffic that is not explicitly permitted on the interface.

- 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 might want to configure ACLs that consist of explicit deny entries, then add an entry to permit all access to the end of each ACL. The permit entry permits packets that are not denied by the deny entries.

Every IPv6 ACL has the following implicit conditions as its last match conditions.

- **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 a **permit ipv6 any any** as the last statement in the access-list if you want to permit IPv6 traffic that were not denied by the previous statements.

  NOTE
  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.

The conditions are applied in the order shown above, with **deny ipv6 any any** as the last condition applied.

For example, if you want to deny ICMP neighbor discovery acknowledgement, 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 entry permits all packets that are not explicitly denied by the other entries. Without this entry, 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 example below.

```
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
```

Creating an IPv6 ACL

Before an IPv6 ACL can be applied to an interface, it must first be created, and then IPv6 must be enabled on that interface.

To create an IPv6 ACL, enter commands such as the following:

```
device(config)# ipv6 access-list fdry
device(config-ipv6-access-list-fdry)# deny tcp host 2001:DB8:e0bb::2 any eq
telnet
device(config-ipv6-access-list-fdry)# permit ipv6 any any
device(config-ipv6-access-list-fdry)# exit
```

This creates an access list that blocks all Telnet traffic from IPv6 host 2001:DB8:e0bb::2.

Syntax for creating an IPv6 ACL

NOTE
The following features are not supported:

- **ipv6-operator flow-label**
- **ipv6-operator fragments** when any protocol is specified. The option **"** fragments**"** can be specified only when "permit/deny ipv6" is specified. If you specify "tcp" or any other protocol instead of "ipv6" the keyword, "fragments" cannot be used.
- **ipv6-operator routing** when any protocol is specified. (Same limitation as for **ipv6-operatorfragments**)

When creating ACLs, use the appropriate syntax below for the protocol you are filtering.

For IPv6 and supported protocols other than ICMP, TCP, or UDP

Syntax: [no] ipv6 access-list *ACL-name*

{ **permit** | **deny** } *protocol*

{ *ipv6-source-prefix***/***prefix-length* | **any** | **host** *source-ipv6_address ipv6-destination-prefix***/***prefix-length* | **any** | **host** *ipv6-destination-address* }

[*ipv6-operator* [*value*]]

[**802.1p-priority-matching** *number*]

[[**dscp-marking** *number* **802.1p-priority-marking** *number* **internal-priority-marking** *number*] | [**dscp-marking** *dscp-value* **dscp-cos-mapping**] | [**dscp-cos-mapping**]]

For ICMP

Syntax: [no] ipv6 access-list *ACL-name*

{ **permit** | **deny** } **icmp** {*ipv6-source-prefix***/***prefix-length* | **any** | **host** *source-ipv6_address ipv6-destination-prefix***/***prefix-length* | **any** | **host** *ipv6-destination-address* }

[*ipv6-operator* [*value*]]

[[*icmp-type*] [*icmp-code*]] | [*icmp-message*]

[**dscp-marking** *number*]

[**dscp-marking** *dscp-value* **dscp-cos-mapping**]

[**dscp-cos-mapping**]]

For TCP

Syntax: [no] ipv6 access-list *ACL-name*

{ **permit** | **deny** } **tcp**

{*ipv6-source-prefix/prefix-length* | **any** | **host** *source-ipv6_address* [*tcp-udp-operator* }

[*source-port-number*]]*ipv6-destination-prefix/prefix-length* | **any** | **host** *ipv6-destination-address* }

[*tcp-udp-operator* [*destination-port-number*]]

[*ipv6-operator* [*value*]]

[**802.1p-priority-matching** *number*]

[**dscp-marking** *number* **802.1p-priority-marking***number* **internal-priority-marking** *number*]

[**dscp-marking** *dscp-value* **dscp-cos-mapping**]

[**dscp-cos-mapping**]

For UDP

Syntax: [no] ipv6 access-list *ACL-name*

{ **permit** | **deny** } *udp*

{*ipv6-source-prefix/prefix-length* | **any** | **host** *source-ipv6_address* [*tcp-udp-operator*

[*source-port-number*]] *ipv6-destination-prefix/prefix-length* | **any** | **host** *ipv6-destination-address* }

[*tcp-udp-operator* [*destination-port-number*]]

[*ipv6-operator* [*value*]]

[**802.1p-priority-matching** *number*]

[**dscp-marking** *number* **802.1p-priority-marking***number* **internal-priority-marking** *number*]

[**dscp-marking** *dscp-value* **dscp-cos-mapping**]

[**dscp-cos-mapping**]

TABLE 15 Syntax descriptions

IPv6 ACL arguments	Description
ipv6 access-list *ACL name*	Enables the IPv6 configuration level and defines the name of the IPv6 ACL. The *ACL name* can contain up to 255 characters and numbers, but cannot begin with a number and cannot contain any spaces or quotation marks. ACL names must be unique across all IPv4 and IPv6 ACLs.
permit	The ACL will permit (forward) packets that match a policy in the access list.
deny	The ACL will deny (drop) packets that match a policy in the access list.
icmp	Indicates the you are filtering ICMP packets.
protocol	The type of IPv6 packet you are filtering. You can specify a well-known name for some protocols whose number is less than 255. For other protocols, you must enter the number. Enter "?" instead of a protocol

TABLE 15 Syntax descriptions (continued)

IPv6 ACL arguments	Description
	to list the well-known names recognized by the CLI. IPv6 protocols include AHP - Authentication Header ESP - Encapsulating Security Payload IPv6 - Internet Protocol version 6 SCTP - Stream Control Transmission Protocol
ipv6-source-prefix/prefix-length	The *ipv6-source-prefix/prefix-length* parameter specify a source prefix and prefix length that a packet must match for the specified action (deny or permit) to occur. You must specify the *ipv6-source-prefix* parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the *prefix-length* parameter as a decimal value. A slash mark (/) must follow the *ipv6-prefix* parameter and precede the *prefix-length* parameter.
ipv6-destination-prefix/prefix-length	The *ipv6-destination-prefix/prefix-length* parameter specify a destination prefix and prefix length that a packet must match for the specified action (deny or permit) to occur. You must specify the *ipv6-destination-prefix* parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the *prefix-length* parameter as a decimal value. A slash mark (/) must follow the *ipv6-prefix* parameter and precede the *prefix-length* parameter
any	When specified instead of the *ipv6-source-prefix /prefix-length* or *ipv6-destination-prefix /prefix-length* parameters, matches any IPv6 prefix and is equivalent to the IPv6 prefix::/0.
host	Allows you specify a host IPv6 address. When you use this parameter, you do not need to specify the prefix length. A prefix length of all128 is implied.
icmp-type	ICMP packets can be filtered by ICMP message type. The type is a number from 0 to 255.
icmp code	ICMP packets, which are filtered by ICMP message type can also be filtered by the ICMP message code. The code is a number from 0 to 255,
icmp-message	ICMP packets are filtered by ICMP messages.
tcp	Indicates the you are filtering TCP packets.
udp	Indicates the you are filtering UDP packets.
ipv6-source-prefix /prefix-length	The *ipv6-source-prefix /prefix-length* parameter specify a source prefix and prefix length that a packet must match for the specified action (deny or permit) to occur. You must specify the *ipv6-source-prefix* parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the *prefix-length* parameter as a decimal value. A slash mark (/) must follow the *ipv6-prefix* parameter and precede the *prefix-length* parameter.
ipv6-destination-prefix /prefix-length	The *ipv6-destination-prefix /prefix-length* parameter specify a destination prefix and prefix length that a packet must match for the specified action (deny or permit) to occur. You must specify the *ipv6-destination-prefix* parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the *prefix-length* parameter as a decimal value. A slash mark (/) must follow the *ipv6-prefix* parameter and precede the *prefix-length* parameter

TABLE 15 Syntax descriptions (continued)

IPv6 ACL arguments	Description
any	When specified instead of the *ipv6-source-prefix /prefix-length* or *ipv6-destination-prefix /prefix-length* parameters, matches any IPv6 prefix and is equivalent to the IPv6 prefix::/0.
host	Allows you specify a host IPv6 address. When you use this parameter, you do not need to specify the prefix length. A prefix length of all128 is implied.
tcp-udp-operator	The *tcp-udp-operator* parameter can be one of the following: • **eq** - The policy applies to the TCP or UDP port name or number you enter after **eq** . • **gt** - The policy applies to TCP or UDP port numbers greater than the port number or the numeric equivalent of the port name you enter after **gt** . Enter **"** ?**"** to list the port names. • **lt** - The policy applies to TCP or UDP port numbers that are less than the port number or the numeric equivalent of the port name you enter after **lt** . • **neq** - The policy applies to all TCP or UDP port numbers except the port number or port name you enter after **neq** . • **range** - The policy applies to all TCP port numbers that are between the first TCP or UDP port name or number and the second one you enter following the **range** parameter. The range includes the port names or numbers you enter. For example, to apply the policy to all ports between and including 23 (Telnet) and 53 (DNS), enter the following **range***23 53* . The first port number in the range must be lower than the last number in the range. The *source-port number* and *destination-port-number* for the tcp-udp-operator is the number of the port.
ipv6-operator	Allows you to filter the packets further by using one of the following options: • **dscp** - The policy applies to packets that match the traffic class value in the traffic class field of the IPv6 packet header. This operator allows you to filter traffic based on TOS or IP precedence. You can specify a value from 0 - 63. • **fragments** - The policy applies to fragmented packets that contain a non-zero fragment offset. **NOTE** This option is not applicable to filtering based on source or destination port, TCP flags, and ICMP flags. • **routing** - The policy applies only to IPv6 source-routed packets. **NOTE** This option is not applicable to filtering based on source or destination port, TCP flags, and ICMP flags.
802.1p-priority-matching *number*	Enables the device to match only those packets that have the same 802.1p priorities as specified in the ACL. Enter 0 - 7. Use this option in conjunction with traffic policies to rate limit traffic for a specified 802.1p priority value.
dscp-marking *number*	Use the **dscp-marking** *number* parameter to specify a new QoS value to the packet. If a packet matches the filters in the ACL statement, this parameter assigns the DSCP value that you specify to the packet. Enter 0 - 63.

TABLE 15 Syntax descriptions (continued)

IPv6 ACL arguments	Description
802.1p-priority-marking *number*	Use the **802.1p-priority-marking** *number* parameter to specify a new QoS value to the packet (0-7). If a packet matches the filters in the ACL statement, this parameter assigns the priority that you specify to the 802.1p priority and the internal priority.
internal-priority-marking *number*	Use the **internal-priority-marking** *number* parameter to specify a new QoS value to the packet (0-7). If a packet matches the filters in the ACL statement, this parameter assigns the priority that you specify to the internal priority and the 802.1p priority. **NOTE** Configuring **802.1p-priority-marking** alone or configuring both **802.1p-priority-marking** and **internal-priority-marking** has the same functionality. That is, it assigns the priority that you specify to the 802.1p priority and the internal priority.
dscp-marking *number*	Use the **dscp-marking** *number* **dscp-cos-mapping** parameters parameters to specify a DSCP value and map that value to an internal QoS table to obtain the packet new QoS value. The following occurs when you use these parameters. • You enter 0 - 63 for the **dscp-marking***number* parameter. • The **dscp-cos-mapping** parameter takes the DSCP value you specified and compares it to an internal QoS table, which is indexed by DSCP values. The corresponding 802.1p priority, internal forwarding priority, and DSCP value is assigned to the packet.

ICMP message configurations

If you want to specify an ICMP message, you can enter one of the following ICMP message types:

- beyond-scope
- destination-unreachable
- echo-reply
- echo-request
- header
- hop-limit
- mld-query
- mld-reduction
- mld-report
- nd-na
- nd-ns
- next-header
- no-admin
- no-route
- packet-too-big
- parameter-option
- parameter-problem

- port-unreachable
- reassembly-timeout
- renum-command
- renum-result
- renum-seq-number
- router-advertisement
- router-renumbering
- router-solicitation
- time-exceeded
- unreachable

NOTE
If you do not specify a message type, the ACL applies to all types ICMP messages types.

Enabling IPv6 on an interface to which an ACL will be applied

Before an IPv6 ACL can be applied to an interface, it must first be created, and then IPv6 must be enabled on that interface.

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 section "IPv6 configuration on each router interface" in the *Brocade FastIron Layer 3 Routing Configuration Guide* .

For example:

```
device(config)#interface ethernet 1/1/1
device(config-if-1/1/1)#ipv6 enable
```

These commands enable IPv6 on Ethernet interface 1/1/1 ready for an IPv6 ACL to be applied.

Syntax for enabling IPv6 on an interface

Syntax: ipv6 enable

When issued at the Interface Configuration level, this command enables IPv6 for a specific interface.

Applying an IPv6 ACL to an interface

As mentioned in IPv6 ACL overview on page 149, IPv6 ACLs are supported on the following devices:

- Gbps Ethernet ports
- 10 Gbps Ethernet ports
- Trunk groups
- Virtual routing interfaces

To apply an IPv6 ACL to an interface, enter commands such as the following.

```
device(config)# interface ethernet 1/3/1
device(config-if-e100-1/3/1)# ipv6 traffic-filter access1 in
```

This example applies the IPv6 ACL "access1" to incoming IPv6 packets on Ethernet interface 1/3/1. As a result, Ethernet interface 1/3/1 denies all incoming packets from the site-local prefix 2001:DB8:0:2::/64 and the global prefix 2001:DB8:1::/48 and permits all other incoming packets.

Syntax for applying an IPv6 ACL

Syntax: ipv6 traffic-filter *ipv6-ACL-name* { **in** | **out** }

For the **ipv6-ACL-name** parameter, specify the name of an IPv6 ACL created using the **ipv6 access-list** command.

The **in** keyword applies the specified IPv6 ACL to incoming IPv6 packets on the interface.

The **out** keyword applies the specified IPv6 ACL to outgoing IPv6 packets on the interface.

Applying an IPv6 ACL to a trunk group

When applying an IPv6 ACL to a trunk group, apply it to the primary port of the trunk, as described under Applying an IPv6 ACL to an interface on page 159. IPv6 ACLs cannot be applied to secondary ports. When an IPv6 ACL is applied to a primary port in a trunk, it filters the traffic on the secondary ports of the trunk as well as the traffic on the primary port.

Applying an IPv6 ACL to a virtual interface in a protocol-based or subnet-based VLAN

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 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.

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
```

Syntax: remark *comment-text*

The *comment-text* can be up to 256 characters in length.

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
```

Syntax: show running-config

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
```

Syntax: [no] remark *comment-text*

For *comment-text*, enter the text exactly as you did when you created the comment.

Support for ACL logging

Brocade devices support ACL logging of inbound packets that are sent to the CPU for processing (denied packets). ACL logging is not supported for any packets that are processed in hardware (permitted packets). ACL logging of both denied as well as permitted outbound packets is not supported.

You may want the software to log entries in the Syslog for inbound packets that are denied by ACL filters. ACL logging is disabled by default; it must be explicitly enabled on a port. Refer to the *ACL logging* section.

Configuring IPv6 ACL accounting

Steps to enable, display, and clear IPv6 ACL accounting

On enabling IPv6 ACL accounting for FastIron devices, it will be enabled on all the filters of the ACL including the implicit rule.

1. To enable IPv6 ACL accounting, use the **enable-accounting** command.

```
device(config-ipv6-access-list v6)#enable-accounting
```

2. To display ACL 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 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 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 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. Here is an example.

```
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
```

Syntax: show ipv6 access-list

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
```

Syntax: show ipv6 access-list [*access-list-name*]

For the **access-list-name** parameter, specify the name of an IPv6 ACL created using the **ipv6 access-list** command.

Policy-Based Routing

Policy-based routing (PBR)

Policy-based Routing (PBR) is the process of altering a packet's path based on criteria other than the destination address. PBR allows you to use ACLs and route maps to selectively modify and route IP packets in hardware. The ACLs classify the traffic. Route maps that match on the ACLs set routing attributes for the traffic.

A PBR policy specifies the next hop for traffic that matches the policy. Using standard ACLs with PBR, you can route IP packets based on their source IP address. With extended ACLs, you can route IP packets based on all of the clauses in the extended ACL.

You can configure the Ruckus device to perform the following types of PBR based on a packet Layer 3 and Layer 4 information:

- Select the next-hop gateway.
- Send the packet to the null interface (null0).

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 are available, the packet is routed in the normal way.

Configuration considerations for policy-based routing

- 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 when IP Follow is configured on an interface.
- Global PBR is not supported with per-port-per-VLAN ACLs.
- A PBR policy on an interface takes precedence over a global PBR policy.
- You cannot apply PBR on a port if that port already has ingress ACLs, ACL-based rate limiting, DSCP-based QoS, 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 six 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. Note that the CLI will allow you configure more than six next hops in a route map; however, the extra next hops will not be placed in the PBR database. The route map could be used by other features like BGP or OSPF, which may use more than six next hops.
- ACLs with the **log** option configured should not be used for PBR purposes.
- PBR ignores explicit or implicit **deny ip any any** ACL entries, to ensure that for route maps that use multiple ACLs, the traffic is compared to all the 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's 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 like TCP/UDP ports, fragmented packed are routed normally.
- You can change route maps or ACL definitions dynamically and do not need to rebind the PBR policy to an interface.

- PBR is supported only on the default VRF.
- PBR is not supported on tunnel interfaces.

Configuring a PBR policy

To configure PBR, you define the policies using IP ACLs and route maps, then enable PBR globally or on individual interfaces. The device programs the ACLs into the packet processor on the interfaces and routes traffic that matches the ACLs according to the instructions in the route maps.

To configure a PBR policy:

- Configure ACLs that contain the source IP addresses for the IP traffic you want to route using PBR.
- Configure a route map that matches on the ACLs and sets the route information.
- Apply the route map on untagged interface or on virtual interface.

Configuring the ACLs

PBR uses route maps to change the routing attributes in IP traffic. This section shows an example of how to configure a standard ACL to identify the source subnet for IP traffic.

To configure a standard ACL to identify a source subnet, enter a command such as the following.

```
device(config)#access-list 99 permit 10.157.23.0 0.0.0.255
```

The command in this example configures a standard ACL that permits traffic from subnet 10.157.23.0/24. After you configure a route map that matches based on this ACL, the software uses the route map to set route attributes for the traffic, thus enforcing PBR.

> **NOTE**
> Do not use an access group to apply the ACL to an interface. Instead, use a route map to apply the ACL globally or to individual interfaces for PBR, as shown in the following sections.

Syntax: [no] access-group *num* { **deny** | **permit** } { *source-ip* | *hostname wildcard* }

or

Syntax: [no] access-list *num* { **deny** | **permit** } { *source-ip/mask-bits* | *hostname* }

Syntax: [no] access-list *num* { **deny** | **permit** } **host** { *source-ip* | *hostname* }

Syntax: [no] access-list *num* { **deny** | **permit** } **any**

The num parameter is the access list number and can be from 1 - 99.

The **deny | permit** parameter indicates whether packets that match a policy in the access list are denied (dropped) or permitted (forwarded).

> **NOTE**
> If you are configuring the ACL for use in a route map, always specify **permit** . Otherwise, the Brocade device will ignore deny clauses and packets that match deny clauses are routed normally.

The source-ip parameter specifies the source IP address. Alternatively, you can specify the host name.

NOTE
To specify the host name instead of the IP address, the host name must be configured using the DNS resolver on the Ruckus device. To configure the DNS resolver name, use the **ip dns server-address** ... command at the global CONFIG level of the CLI.

The wildcard parameter specifies the mask value to compare against the host address specified by the source-ip parameter. 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 to 255, for example 0.0.0.255. Zeros in the mask mean the packet source address must match the source-ip . Ones mean any value matches. For example, the source-ip and wildcard values 10.157.22.26 0.0.0.255 mean that all hosts in the Class C subnet 10.157.22.x match the policy.

If you prefer to specify the wildcard (mask value) in CIDR format, you can enter a forward slash after the IP address, then enter the number of significant bits in the mask. For example, you can enter the CIDR equivalent of "10.157.22.26 0.0.0.255" as "10.157.22.26/24". The CLI automatically converts the CIDR number into the appropriate ACL mask (where zeros instead of ones are the significant bits) and changes the non-significant portion of the IP address into zeros. For example, if you specify 10.157.22.26/24 or 10.157.22.26 0.0.0.255, then save the changes to the startup-config file, the value appears as 10.157.22.0/24 (if you have enabled display of subnet lengths) or 10.157.22.0 0.0.0.255 in the startup-config file.

If you enable the software to display IP subnet masks in CIDR format, the mask is saved in the file in "/ mask-bits " format. To enable the software to display the CIDR masks, enter the **ip show-subnet-length** command at the global CONFIG level of the CLI. You can use the CIDR format to configure the ACL entry regardless of whether the software is configured to display the masks in CIDR format.

NOTE
If you use the CIDR format, the ACL entries appear in this format in the running-config and startup-config files, but are shown with subnet mask in the display produced by the **show ip access-list** command.

The **host** *source-ip* | *hostname* parameter lets you specify a host IP address or name. When you use this parameter, you do not need to specify the mask. A mask of all zeros (0.0.0.0) is implied.

The **any** parameter configures the policy to match on all host addresses.

NOTE
Do not use the **log** option in ACLs that will be used for PBR.

Configuring the route map

After you configure the ACLs, you can configure a PBR route map that matches based on the ACLs and sets routing information in the IP traffic.

NOTE
The **match** and **set** statements described in this section 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.

To configure a PBR route map, enter commands such as the following.

```
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 192.168.2.1
device(config-routemap test-route)# exit
```

The commands in this example configure an entry in a route map named "test-route". The **match** statement matches on IP information in ACL 99. The **set** statement changes the next-hop IP address for packets that match to 192.168.2.1.

To configure a route map without decrementing the Time-to-Live (TTL) value, enter commands such as the following.

```
device(config)# route-map test-route permit 99
device(config-routemap test-route)# match ip address 100
device(config-routemap test-route)# set ip next-hop 192.168.3.1 no-ttl-decrement
device(config-routemap test-route)# exit
```

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.

NOTE
The **no-ttl-decrement** option is supported only on Brocade ICX 7750 and Brocade ICX 7450 devices.

Syntax: [no] route-map *map-name* { **permit** | **deny** } *num*

The *map-name* variable is a string of characters that names the map. Map names can be up to 32 characters in length. You can define an unlimited number of route maps on the Ruckus device, as long as system memory is available.

The **permit | deny** parameter specifies the action the Ruckus device will take if a route matches a match statement:

- If you specify **deny**, the route map instance is ignored and not programmed in Layer 4 CAM.
- If you specify **permit**, the Ruckus device applies the match and set statements associated with this route map instance.

The *num* variable specifies the instance of the route map you are defining. Routes are compared to the instances in ascending numerical order. For example, a route is compared to instance 1, then instance 2, and so on.

When a route map is used in a PBR policy, the PBR policy uses up to six 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.

You can apply multiple ACLs to a route map by entering commands such as the following:

```
device(config)# route-map test-route
device(config-routemap test-route)# match ip address  50 51 52 53 54
```

Syntax: [no] match ip address *ACL-num-or-name*

The *ACL-num-or-name* variable specifies a standard or extended ACL number or name.

Syntax: [no] set ip next-hop *ip-addr* [**no-ttl-decrement**]

The **set ip next-hop** command sets the next-hop IP address for traffic that matches a match statement in the route map. The **no-ttl-decrement** option disables the TTL value decrement and ensures that the packets are forwarded to the neighbor router without decrementing TTL for the matched traffic.

Syntax: [no] set interface null0

The **set interface null0** command sends the traffic to the null0 interface, which is the same as dropping the traffic.

Enabling PBR

After you configure the ACLs and route map entries, you can enable PBR globally, on individual interfaces, or both as described in this section. To enable PBR, you apply a route map you have configured for PBR globally or locally.

Enabling PBR globally

To enable PBR globally, enter a command such as the following at the global CONFIG level.

```
device(config)#ip policy route-map test-route
```

This command applies a route map named "test-route" to all interfaces on the device for PBR.

Syntax: ip policy route-map *map-name*

Enabling PBR locally

To enable PBR locally, enter commands such as the following.

```
device(config)#interface ve 1
device(config-vif-1)#ip policy route-map test-route
```

The commands in this example change the CLI to the Interface level for virtual interface 1, then apply the "test-route" route map to the interface. You can apply a PBR route map to Ethernet ports or virtual interfaces.

Syntax: ip policy route-map *map-name*

Enter the name of the route map you want to use for the **route-map** *map-name* parameter.

Configuration examples for policy based routing

This section presents configuration examples for configuring and applying a PBR policy.

Basic example of policy based routing

The following commands configure and apply a PBR policy that routes HTTP traffic received on virtual routing interface 1 from the 10.10.10.x/24 network to 5.5.5.x/24 through next-hop IP address 1.1.1.1/24 or, if 1.1.1.x is unavailable, through 2.2.2.1/24.

```
deviceBrocade(config)#access-list 101 permit tcp 10.10.10.0 0.0.0.255 eq http 5.5.5.0 0.0.0.255
device(config)#route-map net10web permit 101
device(config-routemap net10web)#match ip address 101
device(config-routemap net10web)#set ip next-hop 1.1.1.1
device(config-routemap net10web)#set ip next-hop 2.2.2.2
device(config-routemap net10web)#exit
device(config)#vlan 10
device(config-vlan-10)#tagged ethernet 1/1/1 to 1/1/4
device(config-vlan-10)#router-interface ve 1
device(config)#interface ve 1
device(config-vif-1)#ip policy route-map net10web
```

Syntax: [no] route-map *map-name* { **permit** | **deny** } *num*

Syntax: [no] set ip next hop*ip-addr*

This command sets the next-hop IP address for traffic that matches a match statement in the route map.

Setting the next hop

The following commands configure the Brocade device to apply PBR to traffic from IP subnets 209.157.23.x, 10.157.24.x, and 209.157.25.x. In this example, route maps specify the next-hop gateway for packets from each of these subnets:

- Packets from 209.157.23.x are sent to 192.168.2.1.
- Packets from 209.157.24.x are sent to 192.168.2.2.

- Packets from 209.157.25.x are sent to 192.168.2.3.

The following commands configure three standard ACLs. Each ACL contains one of the ACLs listed above. Make sure you specify **permit** instead of deny in the ACLs, so that the Brocade device permits the traffic that matches the ACLs to be further evaluated by the route map. If you specify **deny** , the traffic that matches the deny statements are routed normally. Notice that these ACLs specify **any** for the destination address.

```
device(config)#access-list 50 permit 209.157.23.0 0.0.0.255
device(config)#access-list 51 permit 209.157.24.0 0.0.0.255
device(config)#access-list 52 permit 209.157.25.0 0.0.0.255
```

The following commands configure three entries in a route map called "test-route". The first entry (permit 50) matches on the IP address information in ACL 50 above. For IP traffic from subnet 209.157.23.0/24, this route map entry sets the next-hop IP address to 192.168.2.1.

```
device(config)#route-map test-route permit 50
device(config-routemap test-route)#match ip address 50
device(config-routemap test-route)#set ip next-hop 192.168.2.1
device(config-routemap test-route)#exit
```

The following commands configure the second entry in the route map. This entry (permit 51) matches on the IP address information in ACL 51 above. For IP traffic from subnet 209.157.24.0/24, this route map entry sets the next-hop IP address to 192.168.2.2.

```
device(config)#route-map test-route permit 51
device(config-routemap test-route)#match ip address 51
device(config-routemap test-route)#set ip next-hop 192.168.2.2
device(config-routemap test-route)#exit
```

The following commands configure the third entry in the test-route route map. This entry (permit 52) matches on the IP address information in ACL 52 above. For IP traffic from subnet 209.157.25.0/24, this route map entry sets the next-hop IP address to 192.168.2.3.

```
device(config)#route-map test-route permit 52
device(config-routemap test-route)#match ip address 51
device(config-routemap test-route)#set ip next-hop 192.168.2.3
device(config-routemap test-route)#exit
```

The following command enables PBR by globally applying the test-route route map to all interfaces.

```
device(config)#ip policy route-map
test-route
```

Alternatively, you can enable PBR on specific interfaces, as shown in the following example. The commands in this example configure IP addresses in the three source subnets identified in ACLs 50, 51, and 52, then apply route map test-route to the interface.

```
device(config)#interface ve 1
device(config-vif-1)#ip address 209.157.23.1/24
device(config-vif-1)#ip address 209.157.24.1/24
device(config-vif-1)#ip address 209.157.25.1/24
device(config-vif-1)#ip policy route-map test-route
```

Setting the output interface to the null interface

The following commands configure a PBR policy to send all traffic from 192.168.1.204/32 to the null interface, thus dropping the traffic instead of forwarding it.

```
device(config)#access-list 56 permit 192.168.1.204 0.0.0.0
```

The following commands configure an entry in a route map called "file-13". The first entry (permit 56) matches on the IP address information in ACL 56 above. For IP traffic from the host 192.168.1.204/32, this route map entry sends the traffic to the null interface instead of forwarding it, thus sparing the rest of the network the unwanted traffic.

```
device(config)#route-map file-13 permit 56
device(config-routemap file-13)#match ip address 56
device(config-routemap file-13)#set interface null0
device(config-routemap file-13)#exit
```

The following command enables PBR by globally applying the route map to all interfaces.

```
Brocade(config)#ip policy route-map file-13
```

Alternatively, you can enable PBR on specific interfaces, as shown in the following example. The commands in this example configure IP addresses in the source subnet identified in ACL 56, then apply route map file-13 to the interface.

```
device(config)#interface ethernet 1/3/11
device(config-if-e10000-1/3/11)#ip address 192.168.1.204/32
device(config-if-e10000-1/3/11)#ip policy route-map file-13
```

Trunk formation with PBR policy

PBR can be applied on trunk primary port ,only if the port is untagged. When a trunk is formed, the PBR policy on the primary port applies to all the secondary ports. If a different PBR policy exists on a secondary port at the time of a trunk formation, that policy is overridden by the PBR policy on the primary port. If the primary port does not have a PBR policy, then the secondary ports will not have a PBR policy.

When a trunk is removed, the PBR policy that was applied to the trunk interface is unbound (removed) from former secondary ports. If global PBR is configured, the secondary ports adhere to the global PBR; otherwise, no PBR policy is bound to former secondary ports.

Media Access Control Security (MACsec)

MACsec overview

Media Access Control Security (MACsec) is a Layer 2 security technology that provides point-to-point security on Ethernet links between nodes.

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.

Supported MACsec hardware configurations

MACsec key-enabled security can be deployed on a point-to-point LAN between two connected Brocade ICX 7450 devices over interfaces that share a preconfigured static key, the Connectivity Association Key (CAK).

On a licensed Brocade ICX 7450 switch, 10 Gbps ports can be configured for MACsec. Licenses are available per device as described in the *Brocade FastIron Software Licensing Guide*.

NOTE
On the ICX 7450, MACsec is available only on 4 X 10GF modules present in slots 2, 3, or 4.

NOTE
MACsec on ICX devices can interoperate with MACsec on MLXE devices.

MACsec RFCs and standards

FastIron MACsec is one of several IEEE 802.1X capabilities supported by Brocade Ethernet switches.

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 the “MAC Port Security” section 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.
- On the ICX 7450 switch, ports on a 4X10GF removable module installed in device module 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 *Brocade 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 Brocade 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 typical 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, currently Galois/Counter Mode Advanced Encryption Standard 128 (GCM-AES-128), 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.

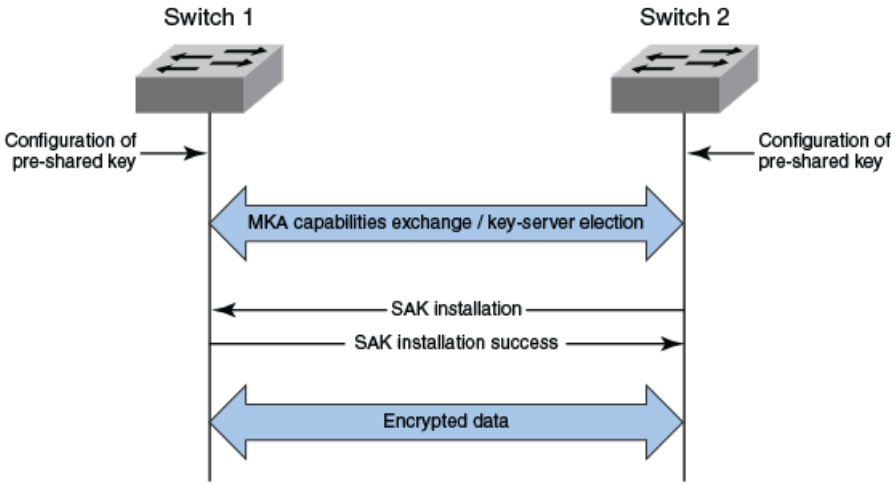
MKA message exchange between two switches

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.

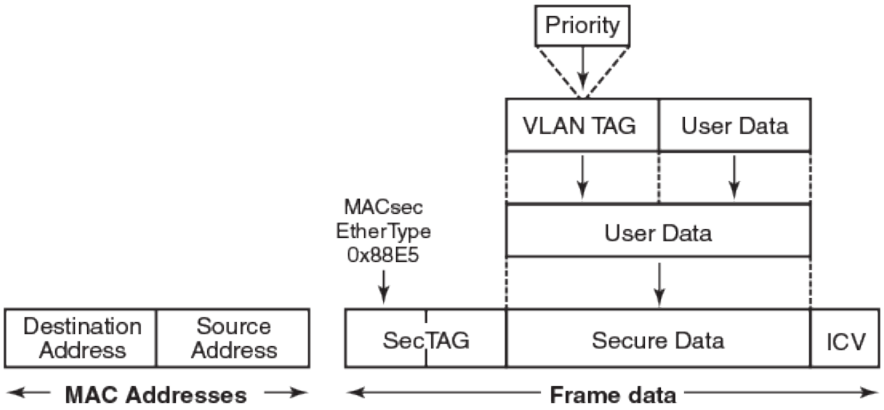
The secure association is designated by a Secure Association Identifier (SAI), formed from the Secure Channel Identifier (SCI) combined with an Association Number (AN). When a MACsec frame is received by a peer interface, the Brocade device identifies the session key from the SAI carried in the MACsec frame and uses the key to decrypt and authenticate the received frame.

MACsec frame format

When MACsec is enabled, Brocade 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.

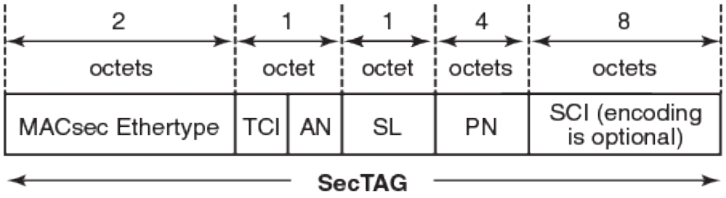
FIGURE 2 MACsec frame format



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

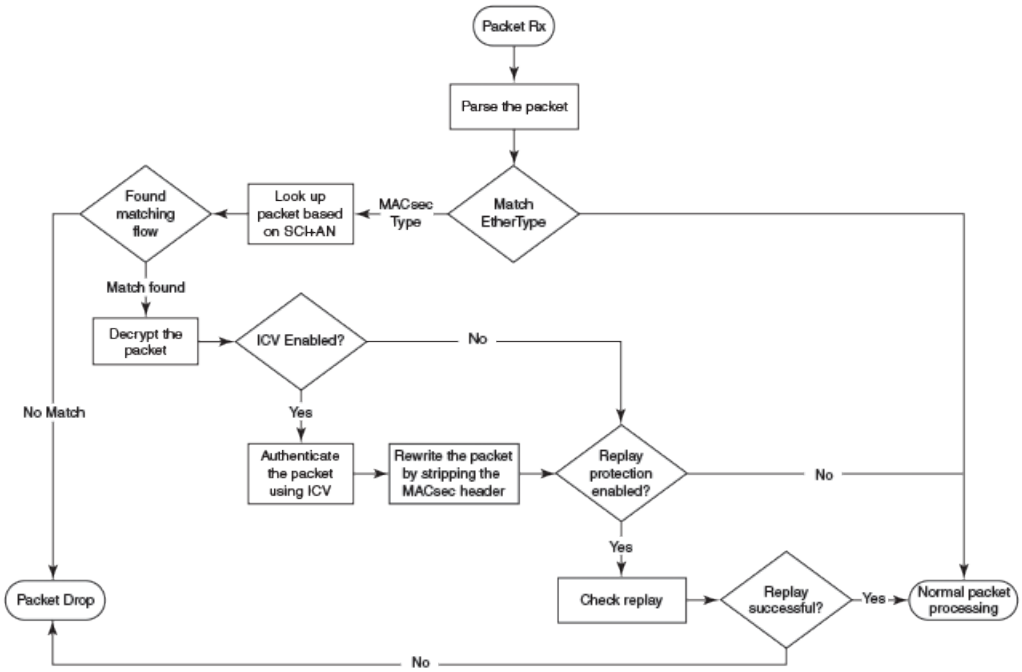


Processing incoming frames

Brocade hardware processes each MACsec frame received or transmitted based on the information in the MACsec security tag.

The Brocade switch 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

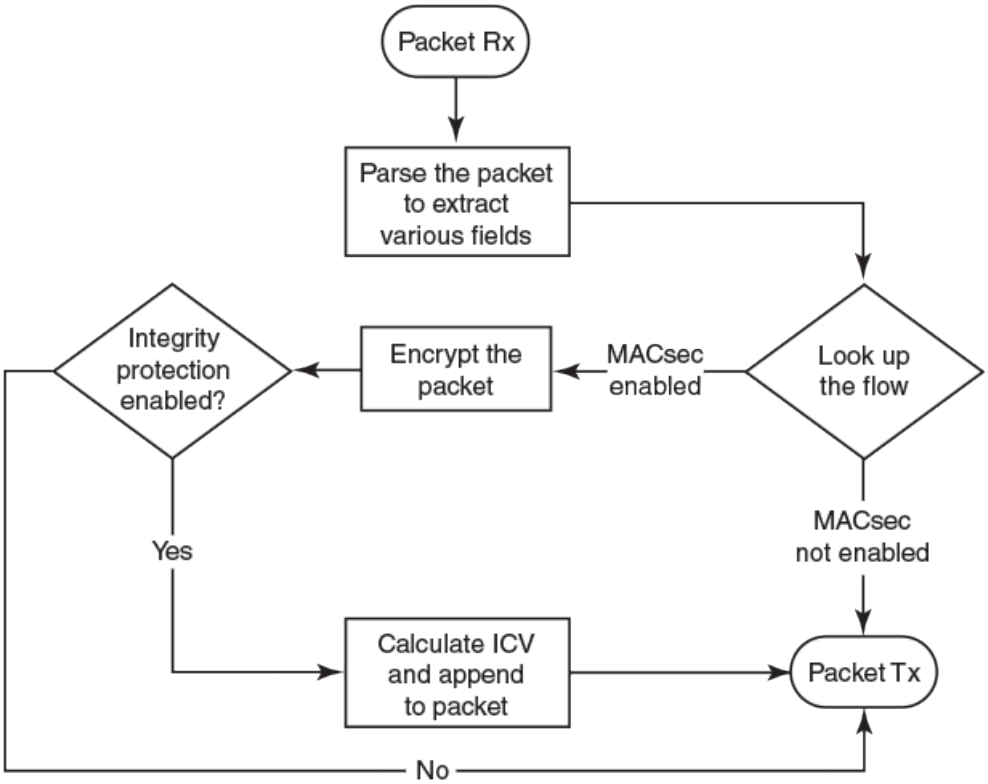


Processing outgoing frames

The Brocade switch 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 switch 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 command reference page for each command for default values.

6. Configure 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.

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 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 178 for an overview of enabling and configuring MACsec features.

1. Use the following command to enter global configuration mode.

```
device# configure terminal
```

2. Use the following command to enable MKA capabilities and enters dot1x-mka configuration mode.

```
device(config)# dot1x-mka-enable
```

3. Use the following command to enter dot1x-mka group configuration mode.

```
device(config-dot1x-mka)# mka-cfg-group group1
```

4. 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(config-dot1x-mka-group-group1)# key-server-priority 20
```

In this example, the key-server priority is set to 20 for the MKA group group1.

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 (GCM-AES-128) 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 178 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.

   In the following example, MACsec encryption has been configured as a group test1 setting. By default, ICV integrity check is also enabled.

```
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 178 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 178 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.
- disable: Frames are not validated.

NOTE
The disable option is a duplicate option available on the ICX 7450 switch. 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 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 ethernet** 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)# mka-cfg-group test1
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)# mka-cfg-group test1
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 178 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)# mka-cfg-group test1
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(config)# dot1x-mka
  dot1x-mka-enable              Enable MACsec
device(config)# dot1x-mka-enable
device(config-dot1x-mka)#
device(config-dot1x-mka)# mka-cfg-group
  ASCII string   Name for this group
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)#

device(config-dot1x-mka-group-test1)# key-server-priority
  DECIMAL   Priority of the Key Server. Valid values should be between 0 and 255
device(config-dot1x-mka-group-test1)# key-server-priority 5
device(config-dot1x-mka-group-test1)#

device(config-dot1x-mka-group-test1)# macsec cipher-suite
  gcm-aes-128   GCM-AES-128 Cipher suite
```

```
device(config-dot1x-mka-group-test1)# macsec cipher-suite gcm-aes-128
device(config-dot1x-mka-group-test1)#

device(config-dot1x-mka-group-test1)# macsec confidentiality-offset
  30   Confidentiality offset of 30
  50   Confidentiality offset of 50
device(config-dot1x-mka-group-test1)# macsec confidentiality-offset 30
device(config-dot1x-mka-group-test1)#

device(config-dot1x-mka-group-test1)# macsec frame-validation
  check     Validate frames with secTAG and accept frames without secTAG
  disable   Disable frame validation
  strict    Validate frames with secTAG and discard frames without secTAG
device(config-dot1x-mka-group-test1)# macsec frame-validation strict
device(config-dot1x-mka-group-test1)#

device(config-dot1x-mka-group-test1)# macsec replay-protection
  out-of-order   Validate MACsec frames arrive in the given window size
  strict         Validate MACsec frames arrive in a sequence
device(config-dot1x-mka-group-test1)# macsec replay-protection strict
device(config-dot1x-mka-group-test1)#

device(config-dot1x-mka)#enable-mka e 1/3/2
device(config-dot1x-mka-1/3/2)#

device(config-dot1x-mka-1/3/2)# mka-cfg-group
  ASCII string   Name for the group to be applied
device(config-dot1x-mka-1/3/2)# mka-cfg-group test1
device(config-dot1x-mka-1/3/2)#

device(config-dot1x-mka-1/3/2)# pre-shared-key 135bd758b0ee5c11c55ff6ab19fdb199 key-name
96437a93ccf10d9dfe347846cce52c7d
device(config-dot1x-mka-1/3/2)#
```

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. In addition, **show** commands are available 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. At the EXEC or Privileged EXEC level, 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-dot1x-mka-1/3/3)# 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. At the EXEC or Privileged EXEC level, 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-dot1x-mka-1/3/3)# 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.

```
device(config-dot1x-mka-1/3/3)# 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, at the EXEC or Privileged EXEC level, enter the **show dot1x-mka sessions ethernet** command followed by the interface identifier.

```
device(config-dot1x-mka-1/3/3)# 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.

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. At the EXEC or Privileged EXEC level, enter the **clear macsec statistics ethernet** command for the designated interface.

   Results of the previous **show macsec ethernet** command are removed.

2. Enter the **show macsec statistics ethernet** command to display information on MACsec configuration and secure channel activity for a particular interface.

   The following **show macsec statistics ethernet** command output is for an ICX 7450.

```
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 feature allows you to configure the device to learn a limited number of secure MAC addresses on an interface. The interface forwards only packets with source MAC addresses that match these secure addresses.

The secure MAC addresses can be specified statically or learned dynamically. If the device reaches the maximum limit for the number of secure MAC addresses allowed on the interface and if the interface receives a packet with a source MAC address that is different from any of the secure learned addresses, it is considered a security violation.

When a security violation occurs, a Syslog entry and an SNMP trap are generated. In addition, the device takes one of two actions: it either drops packets from the violating address (and allows packets from the secure addresses), or disables the port for a specified amount of time. You specify which of these actions takes place.

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 MAC port security

The MAC port security feature uses a concept of local and global "resources" to determine how many MAC addresses can be secured on each interface. In this context, a "resource" is the ability to store one secure MAC address entry. Each interface is allocated 64 local resources. Additional global resources are shared among all interfaces on the device.

When the MAC port security feature is enabled on an interface, the interface can store one secure MAC address. You can increase the number of MAC addresses that can be secured using local resources to a maximum of 64.

Besides the maximum of 64 local resources available to an interface, there are additional global resources. Depending on flash memory size, a device can have 1024, 2048, or 4096 global resources available. When an interface has secured enough MAC addresses to reach its limit for local resources, it can secure additional MAC addresses by using global resources. 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.

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.

Enabling port MAC security globally

Port MAC security can be enabled at the device level. Autosaving of secure MAC addresses and setting a port security age timer can also be configured at the device level.

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 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.

5. 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.

In the following example port MAC security is globally enabled on a device, secure MAC addresses are saved to the startup configuration every 20 minutes and timed out after 60 minutes.

```
device# configure terminal
device(config)# port security
device(config-port-security)# enable
device(config-port-security)# age 60 absolute
device(config-port-security)# autosave 20
```

Enabling port MAC security on a specific interface

Port MAC security can be enabled for a specific interface on a device.

In addition to enabling port security on a specified interface, the maximum number of secured MAC addresses can be configured. A secure MAC address is configured in port MAC security interface configuration mode.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode.

```
device(config)# interface ethernet 1/7/11
```

3. Enter port MAC security configuration mode.

```
device(config-if-e1000-1/7/11)# port security
```

4. Enable port MAC security on the interface.

```
device(config-port-security-e1000-1/7/11)# enable
```

5. Set the maximum number of secure MAC addresses for the interface.

```
device(config-port-security-e1000-1/7/11)# maximum 10
```

6. Specify a secure MAC address.

```
device(config-port-security-e1000-1/7/11)# secure-mac-address 000.0018.747c
```

   In this example, a secure MAC address is specified on the untagged Ethernet interface 1/7/11.

In the following example port MAC security is enabled on an untagged Ethernet interface device, up to ten secure MAC addresses can be specified on this interface, and a secure MAC address is specified.

```
device# configure terminal
device(config)# interface ethernet 1/7/11
device(config-if-e1000-1/7/11)# port security
device(config-port-security-e1000-1/7/11)# enable
device(config-port-security-e1000-1/7/11)# maximum 10
device(config-port-security-e1000-1/7/11)# secure-mac-address 000.0018.747c
```

Specifying the action taken when a security violation occurs

A security violation can occur when a user tries to connect to a port where a MAC address is already locked, or the maximum number of secure MAC addresses has been exceeded. When a security violation occurs, an SNMP trap and Syslog message are generated.

You can configure the device to take one of two actions when a security violation occurs; either drop packets from the violating address (and allow packets from secure addresses), or disable the port for a specified time.

Dropping packets from a violating address

To configure the device to drop packets from a violating address and allow packets from secure addresses, enter the following commands.

```
device(config)#interface ethernet 1/7/11
device(config-if-e1000-1/7/11)#port security
device(config-port-security-e1000-1/7/11)#violation restrict
```

Syntax: violation [**restrict**]

NOTE
When the **restrict** option is used, the maximum number of MAC addresses that can be restricted is 128. If the number of violating MAC addresses exceeds this number, the port is shut down. An SNMP trap and the following Syslog message are generated: "Port Security violation restrict limit 128 exceeded on interface ethernet *port_id* ". This is followed by a port shutdown Syslog message and trap.

Specifying the period of time to drop packets from a violating address

To specify the number of minutes that the device drops packets from a violating address, use commands similar to the following.

```
device(config)#interface ethernet 1/7/11
device(config-if-e1000-1/7/11)#port security
device(config-port-security-e1000-1/7/11)#violation restrict 5
```

Syntax: violation [**restrict**] [*age*]

The age variable can be from 0 through 1440 minutes. The default is 5 minutes. Specifying 0 drops packets from the violating address permanently.

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.

Disabling the port for a specified amount of time

You can configure the device to disable the port for a specified amount of time when a security violation occurs.

To shut down the port for 5 minutes when a security violation occurs, enter the following commands.

```
device(config)#interface ethernet 1/7/11
device(config-if-e1000-1/7/11)#port security
device(config-port-security-e1000-1/7/11)#violation shutdown 5
```

Syntax: violation [**shutdown**] [*minutes*]

The minutes can be from 0 through 1440 minutes. Specifying 0 shuts down the port permanently when a security violation occurs.

Clearing port security statistics

You can clear restricted MAC addresses and violation statistics from ports on all ports or on 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 a command such as the following.

```
Brocade#clear port security restricted-macs ethernet 5
```

Syntax: clear port security restricted-macs [**all** | **ethernet** *port*]

Clearing violation statistics

To clear violation statistics globally, enter the **clear port security statistics** all command.

```
device#clear port security statistics all
```

To clear violation statistics on a specific port, enter a command such as the following.

```
device#clear port security statistics ethernet 1/1/5
```

Syntax: clear port security statistics [**all** | **ethernet** *port*]

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.

```
Brocade# 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 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 filters command syntax

To configure and apply a MAC address filter, enter commands such as the following.

```
device(config)# mac filter 1 deny 0000.0075.3676 ffff.0000.0000
device(config)# mac filter 2 deny any ffff.ffff.ffff 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)# int e 1
device(config-if-e1000-1)# mac filter-group 1 to 5 1024
```

These commands configure filter 1 to deny traffic with a source MAC address that begins with "3565" to any destination, and configure filters 2 through 5 to deny traffic with the specified destination MAC addresses. Filter 1024 permits all traffic that is not denied by any other filter.

NOTE
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.

Syntax: [no] mac filter *filter-num* { **permit** | **deny** } [*src-mac mask* | **any**] [*dest-mac mask* | **any**]

You can configure up to 507 MAC filters for *filter-num*. The default value is 512.

The **permit or deny** argument determines the action the software takes when a match occurs.

The **src-mac mask | any** parameter specifies the source MAC address. You can enter a specific address value and a comparison mask or the keyword **any** to filter on all MAC addresses. Specify the mask using f (ones) and zeros. For example, to match on the first two bytes of the address aabb.ccdd.eeff, use the mask ffff.0000.0000. In this case, the filter matches on all MAC addresses that contain "aabb" as the first two bytes. The filter accepts any value for the remaining bytes of the MAC address. If you specify **any** , do not specify a mask. In this case, the filter matches on all MAC addresses.

The **dest-mac mask | any** parameter specifies the destination MAC address. The syntax rules are the same as those for the **src-mac mask | any** parameter.

Syntax: no mac filter log-enable

Globally enables logging for filtered packets.

Syntax: no mac filter-group log-enable

Enables logging for filtered packets on a specific port.

Syntax: [no] mac filter-group *filter-number* [**to** *filter-number* | *filter-number ...*]

Applies MAC address filters to a port.

When applying the filter-group to the interface, specify each line to be applied separately or use the **to** keyword to apply a consecutive range of filter lines, for example, 1 3 to 8 10.

NOTE
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.

NOTE
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.

NOTE
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.

When a MAC address filter is applied to or removed from an interface, a Syslog message such as the following 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.

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 command syntax

To configure MAC address filter logging globally, enter the following CLI commands at the global CONFIG level.

```
device(config)#mac filter log-enable
device(config)#write memory
```

Syntax: [no] mac filter log-enable

To configure MAC address filter logging for MAC address filters applied to ports 1 and 3, enter the following CLI commands.

```
device(config)#int ethernet 1
device(config-if-e1000-1)#mac filter-group log-enable
device(config-if-e1000-1)#int ethernet 3
device(config-if-e1000-3)#mac filter-group log-enable
device(config-if-e1000-3)#write memory
```

Syntax: [no] mac filter-group log-enable

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

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 (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.

NOTE
The term "client" is used to indicate the user or device that is going through authentication.

Brocade FastIron devices support the IEEE 802.1X standard for authenticating clients attached to LAN ports. Using 802.1X, you can configure a FastIron 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 Brocade 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.

MAC authentication is a mechanism by which incoming traffic originating from a specific MAC address is switched or forwarded by the device only if the source MAC address is successfully authenticated by an authentication server. 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 network resources are accessed by the client. VLAN policies and ACL policies can be enforced using VLAN assignment and ACL assignment to provide different levels of services to the client and to control the destination of the client.

NOTE
Flexible authentication combines MAC authentication and 802.1X authentication as a single authentication procedure. Refer to *Brocade FastIron Features and Standards Support Matrix* for the list of supported platforms.

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 various 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 (MAC address of the client), not to the entire port.

When authentication succeeds, the client is moved to the VLAN returned by the RADIUS server.

NOTE
A system default VLAN, reserved VLANs, or VLAN groups cannot be used as the auth-default-VLAN, RADIUS-assigned VLAN, restricted VLAN, critical VLAN and guest 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 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 attempt 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 configured restricted VLAN instead of blocking the client completely. The port is moved to the configured restricted VLAN only if the authentication failure action is set to place the port in a restricted VLAN using the **auth-fail-action** command from the authentication configuration mode or using the **authentication fail-action** command at the interface level. Otherwise, when the authentication fails, the client's MAC address is blocked in the hardware (default action).
- Critical VLAN: There may be scenarios in which the RADIUS server times out or is not available, resulting in authentication failure. This can happen the first time the client is authenticating or when the client re-authenticates. In such scenarios, if the authentication timeout action is specified as a critical VLAN using the **authentication timeout-action** command from the interface configuration mode, the client is moved to the specified critical VLAN. A critical VLAN can be configured using the **critical-vlan** command from 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 authenticator loaded. In such a scenario, the client is moved to the guest VLAN to have access to the network with default privileges, from where the client can download the authenticator.

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.

After successful authentication (Flexible authentication), VLANs are dynamically assigned based on the user or device profiles configured on the RADIUS server. The switch then associates the port with the dynamic VLAN only for the particular user or device 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 or devices are untagged and are classified into appropriate VLANs when they are received on the switch. This makes the port look as if it is part of multiple tagged VLANs. In summary, after successful authentication, the RADIUS server returns the details of the VLAN where the client should belong. The client (MAC address of the client) is moved to this VLAN as a MAC VLAN member. The client is removed from the corresponding VLAN in situations when the client logs out, the port goes down, or when the MAC address ages out.

Authentication success and failure actions

With Flexible authentication, success and failure actions are applied for MAC authentication and 802.1X authentication.

Authentication success action

When the authentication sequence is set to perform 802.1X authentication followed by MAC authentication (default Flexible authentication sequence), upon 802.1X authentication, the client is authenticated and the policies returned by the RADIUS server are applied.

When the authentication sequence is set to perform MAC authentication followed by 802.1X authentication, by default, 802.1X authentication is performed even if MAC authentication is successful. Upon successful 802.1X authentication, the client is authenticated and the policies returned by the RADIUS server are applied.

Authentication failure action

You can define a single failure action 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 (default action): This blocks the client from accessing any network resource for a configured amount of time, after which it can try authenticating again.
- Move the client to a restricted VLAN: This moves the client to a preconfigured restricted VLAN. Any access policies applied in that VLAN apply to this client. Reauthentication in a restricted VLAN is set by the **authentication reauth-timeout** command at the interface level. The timeout is enabled by default and set to 60 seconds.

Authentication timeout actions

A single authentication timeout action can be specified for MAC authentication and 802.1X authentication timeouts.

A RADIUS timeout occurs when the Brocade device does not receive a response from a RADIUS server within a specified time limit 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 respectively. If the parameters are not manually configured, the Brocade device applies the default value of 3 seconds with a maximum of 3 retries.

You can better control port behavior when a RADIUS timeout occurs by configuring a port on the Brocade device to automatically pass or fail user authentication. A pass bypasses the authentication process and allows the client to continue with the VLAN and

other policies. A fail blocks the client by default, unless a restricted VLAN is configured, in which case, the user is placed into a VLAN. By default, the Brocade device resets the authentication process and retries to authenticate the user.

The following options are available:

- Failure (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: When the RADIUS timeout action is configured as "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.
- Move the client to a critical VLAN: The client is moved to a preconfigured critical VLAN. Any access policies applied to that VLAN will apply to this client.

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 interface level. The timeout is enabled by default and is set to 60 seconds.

NOTE
Reauthentication is supported for restricted and critical VLANs. It is not supported for guest VLANs.

RADIUS attributes for authentication

RADIUS attributes are used to define specific authentication, authorization, and accounting (AAA) elements in a user profile, which is stored on the RADIUS daemon. When a client successfully completes the EAP authentication process, the authentication server (the RADIUS server) sends the authenticator (the Brocade device) a RADIUS Access-Accept message that grants the client access to the network. The RADIUS Access-Accept message contains attributes set for the user in the user's access profile on the RADIUS server. Many functions, such as dynamic VLAN assignment, dynamic IP ACL and MAC filter assignment, and authentication sequence rules for Flexible authentication, are based on the RADIUS attributes. Ruckus devices support the following RADIUS attributes for 802.1X authentication and MAC authentication:

- Username (1) - RFC 2865
- NAS-IP-Address (4) - RFC 2865
- NAS-Port (5) - RFC 2865
- Service-Type (6) - RFC 2865
- Filter-Id (11) - RFC 2865
- Framed-MTU (12) - RFC 2865
- State (24) - RFC 2865
- Vendor-Specific (26) - RFC 2865
- Session-Timeout (27) - RFC 2865
- Termination-Action (29) - RFC 2865
- Calling-Station-ID (31) - RFC 2865
- NAS-Identifier (32) - RFC 2865
- NAS-Port-Type (61) - RFC 2865
- Tunnel-Type (64) - RFC 2868
- Tunnel-Medium-Type (65) - RFC 2868

- EAP Message (79) - RFC 2579 (Only for 802.1X authentication)
- Message-Authenticator (80) RFC 3579
- Tunnel-Private-Group-Id (81) - RFC 2868
- NAS-Port-id (87) - RFC 2869

Configuring Ruckus-specific attributes on the RADIUS server

If the RADIUS authentication process is successful, the RADIUS server sends an Access-Accept message to the Ruckus device, authenticating the device. The Access-Accept message can include vendor-specific attributes (VSAs) that specify additional information about the device. If you are configuring MAC authentication and 802.1X authentication on the same port, then you can configure the Ruckus VSAs listed in following table on the RADIUS server.

You add the Ruckus vendor-specific attributes to your RADIUS server configuration, and configure the attributes in the individual or group profiles of the devices that will be authenticated. The Ruckus Vendor-ID is 1991, with Vendor-Type 1. For more information, refer to Configuring RADIUS on page 63.

TABLE 16 Ruckus vendor-specific attributes 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. Set the attribute to 0 for devices that do not support 802.1X authentication. **1** - Perform 802.1X authentication when a device passes MAC authentication. Set the attribute to 1 for devices that support 802.1X 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 authentication and 802.1X authentication.

These VSAs can be used in a device profile on the RADIUS server for MAC authentication. These VSAs are optional. These VSAs are only applicable when both MAC authentication and 802.1X authenticaion are configured on the port and authentication sequence is MAC authentication followed by 802.1X authentication. These VSAs are not needed in the device profile if only MAC authentication is enabled on the port.

LLDP and CDP parameters for IP phones using RADIUS attributes

You can add the Brocade vendor-specific attribute "Foundry-Voice-Phone-Config" in the RADIUS server configuration to identify whether the client is a voice or phone device. You can also specify the voice or phone device configuration if required.

If the switch receives an attribute that identifies the incoming client as a voice or phone device, Link Layer Discovery Protocol (LLDP) and Cisco Discovery Protocol (CDP) protocol information are automatically configured. LLDP and CDP advertise the VLAN information so that the client connected to the port learns the voice VLAN. CDP does not require any specific information, whereas LLDP requires differentiated services code point (DSCP) and priority values to configure the Media Endpoint Discovery (MED) policy. These two values can be optionally specified in the VSA, otherwise default values of dscp(46), priority(5) are applied. Possible values for the VSA are " " with a space between the quotation marks (which sets the default values), "dscp: 40; priority:4", "dscp:30", "priority:7", and so on.

TABLE 17 Ruckus vendor-specific attributes for RADIUS to identify the client as phone device

Attribute name	Attribute ID	Data type	Description
Foundry-Voice-Phone-Config	11	string	Identifies the client as a voice or phone device and optionally specifies the voice or phone device configuration.

Support for the RADIUS user-name attribute in Access-Accept messages

Authentication-enabled ports support the RADIUS user-name (type 1) attribute in the Access-Accept message returned during authentication.

In the case of 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.

When you enable sFlow forwarding 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 *Brocade 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 dot1x** commands, such as **show dot1x sessions**.

This same information is sent as the user-name attribute of RADIUS accounting messages, and is sent to the RADIUS accounting servers.

To enable the user-name attribute, add the following attribute on the RADIUS server.

Attribute name	Type	Value
user-name	1	*name* (string)

Flexible authentication with dynamic VLAN assignment

After successful authentication, a VLAN assignment policy can be applied to control the destination of the client. Dynamic VLAN assignment allows clients to connect to the network anywhere and, based on their credentials, they get placed in the correct 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 attribute sent from RADIUS server.

A client can be dynamically assigned to a VLAN based on the attribute sent from the RADIUS server. When a client is successfully authenticated, the RADIUS server sends the Brocade device a RADIUS Access-Accept message that allows the Brocade device to forward traffic from that client (MAC address of the client). To enable dynamic VLAN assignment for authenticated clients, you must add attributes to the profile for the client on the RADIUS server. Because Flexible authentication uses dynamic VLANs, it is recommended to configure the VLAN information in the RADIUS server. Refer to Configuring the RADIUS server to support dynamic VLAN assignment for authentication on page 207 for a list of the attributes that must be set on the RADIUS server.

If one of the attributes in the Access-Accept message specifies a VLAN identifier, and the VLAN is available on the Brocade device, the port becomes a MAC VLAN member of the specified VLAN.

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. If the authentication sequence is configured to perform MAC authentication followed by 802.1X authentication, 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, and a combination of tagged and untagged VLANs for different use cases; for example, client devices such as computers, IP phones, wireless access points, servers running hypervisors with mulitple Virtual Machines (VMs), and so on.

To specify VLAN identifiers on the RADIUS server, add the following attributes to the device profile for MAC authentication. For 802.1X authentication, add these attributes to the user profile.

TABLE 18 Attributes for dynamic VLAN assignment

Attribute name	Type	Value
Tunnel-Type	064	13 (decimal) - VLAN
Tunnel-Medium-Type	065	6 (decimal) - 802
Tunnel-Private-Group-ID	081	vlan-id vlan-name U:vlan-id U:vlan-name T:vlan-name T:vlan-id 1; T:vlan-id 2

NOTE
Different formats are supported for Tunnel-Private-Group-ID in different FastIron releases. FastIron 8.0.30b and later releases support the following formats: vlan-id, U: , T:, U: T:, multiple T:, and U: multiple T: .

The device reads 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 Ruckus device ignores these Attribute-Value pairs. If the Tunnel-Private-Group-ID is valid, then the client gets authorized in this VLAN, otherwise it will be authorized in the auth-default VLAN.
- When the Ruckus device receives the value specified for the Tunnel-Private-Group-ID attribute, it checks whether the *vlan-name* string matches the name of a VLAN configured on the device. If there is a VLAN on the device with a name that matches the *vlan-name* string, then the client port is placed in the VLAN with an ID that corresponds to the VLAN name.
- If the *vlan-name* string does not match the name of a VLAN, the Ruckus device checks whether the string, when converted to a number, matches the ID of a VLAN configured on the device. If the ID matches, then the client port is placed in the VLAN with that ID.
- If the *vlan-name* string does not match either the name or the ID of a VLAN configured on the device, then the client will not become authorized.

Single and multiple untagged VLAN mode

In 802.1X authentication and MAC authentication, by default, a port will be a member of only one untagged VLAN (single untagged mode).The single untagged mode is the most common use case and in a scenario in which 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 dynamic untagged VLANs, subsequent clients are blocked.

In multiple untagged mode, different clients on the same port can be placed into different untagged VLANs. The authentication-enabled ports or a specific port can be configured to be a member of multiple untagged VLANs (multiple untagged mode) using the **auth-vlan-mode** command at the global level or using the **authentication auth-vlan-mode** command from the interface configuration mode.

NOTE
Untagged packets from tagged client are not allowed in multiple untagged mode.

Dynamic assignment of VLANs in the single and multiple untagged modes varies depending on the format of the VLAN information in the RADIUS-returned Access-Accept message and on whether the authentication is initiated by tagged or untagged ports. For more information, refer to Dynamic VLAN assignment in authentication success scenarios on page 208 and Dynamic VLAN assignment in authentication failure scenarios on page 210.

Dynamic VLAN assignment in authentication success scenarios

The dynamic VLAN assignment depends on the various VLAN formats returned by the RADIUS server.

Dynamic VLAN assignment when RADIUS returns only VLAN identifier

When the Access-Accept message returned by RADIUS contains the VLAN information in either *vlan-id* or *vlan-name* format:

- By default (single untagged mode), the port membership is removed from the auth-default VLAN and added to the RADIUS-specified VLAN as a MAC VLAN member (single untagged mode).

  Following is the default behavior of the device (in single untagged mode):

  - Subsequent clients that are authenticated with different dynamic VLANs are blocked.
  - If another client is authenticated on the same VLAN as that of the first, the new client is permitted in the first client's dynamic VLAN.

  - If another client is authenticated on the port without a RADIUS VLAN, the new client is permitted in the first clients' 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.

Dynamic VLAN assignment when RADIUS returns untagged VLAN identifier

When the Access-Accept message returned by RADIUS contains the VLAN information in U:*vlan-id* format:

- By default (single untagged mode), the port membership is removed from the auth-default VLAN and added to the RADIUS-specified VLAN as a MAC VLAN member.

  Following is the default behavior of the device (in single untagged mode):

  - Subsequent clients that are authenticated with different dynamic VLANs are blocked.
  - If another client is authenticated on the same VLAN as that of the first, the new client is permitted in the first client's dynamic VLAN.
  - If another client is authenticated on the port without a RADIUS VLAN, the new client is permitted in the first clients' 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.

Dynamic VLAN assignment when RADIUS returns a tagged VLAN identifier 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:

In the case of MAC authentication, if the authentication is triggered by a tagged packet, and if its 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.

Dynamic VLAN assignment when RADIUS returns untagged and tagged VLAN identifiers

When the Access-Accept message returned by RADIUS contains the VLAN information in U:*vlan-id1*; T:*vlan-id2* format:

- By default (single untagged mode), the port membership is removed from the auth-default VLAN and added to the RADIUS-specified VLAN as a MAC VLAN member.

  Following is the default behavior of the device (in single untagged mode):

  - Subsequent clients that are authenticated with different dynamic VLANs are blocked.
  - If another client is authenticated on the same VLAN as that of the first, the new client is permitted in the first client's dynamic VLAN.
  - If another client is authenticated on the port without a RADIUS VLAN, the new client is permitted in the first clients' 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.

Dynamic VLAN assignment when RADIUS returns untagged and multiple tagged VLAN identifiers

When the Access-Accept message returned by RADIUS contains the VLAN information in U:*vlan-id1*;T:*vlan-id2*;T:*vlan-id3*;T:*vlan-id4* format:

- By default (single untagged mode), the port membership is removed from the auth-default VLAN and added to the RADIUS-specified VLAN as a MAC VLAN member.

  Following is the default behavior of the device (in single untagged mode):

  - Subsequent clients that are authenticated with different dynamic VLANs are blocked.
  - If another client is authenticated on the same VLAN as that of the first, the new client is permitted in the first client's dynamic VLAN.
  - If another client is authenticated on the port without a RADIUS VLAN, the new client is permitted in the first clients' 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.

In the case of MAC authentication, if the authentication is triggered by a tagged packet, and if its 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 (default), if an authenticated client exists on a port and the second client trying to authenticate fails, then the port is not moved to a restricted VLAN. The client that failed authentication is blocked.

Dynamic VLAN assignment in authentication failure scenarios

VLAN assignment for authentication failure actions specified for a client varies. VLAN assignment depends on whether the Flexible authentication-enabled port is configured to be a member of only one untagged VLAN (single untagged mode; default behavior) or whether the port is a member of multiple untagged VLANs (multiple untagged mode). You can configure a single authentication failure action that applies to MAC authentication and 802.1X authentication. The authentication failure action can be one of the following:

- Block the client's MAC address
- Move the client to a restricted VLAN

Authentication failure actions in single untagged mode

If the failure action is configured as a restricted VLAN:

- If the first client's authentication fails, the port's membership is moved from the auth-default VLAN to the restricted VLAN.
- If other clients were authenticated previously on the same port, the MAC address of the new client is blocked. Even after all other clients age out, the new client remains in the VLAN reserved for blocked clients until it ages out.
- If a failure action is not configured, the client's MAC address is blocked.
- If the previous sessions are in a restricted VLAN, the existing MAC sessions are cleared before permitting a new client in the auth-default VLAN or RADIUS-specified VLAN.
- If the previous sessions are in the critical VLAN or guest VLAN, the MAC address of the new client is blocked.
- In the case of 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 failure actions in multiple untagged mode

- 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 an untagged member of the restricted VLAN.
- If a failure action is not configured, the client's MAC address is blocked.
- In the case of MAC authentication, if the authentication is initiated by tagged packet, the client is blocked in the tagged VLAN irrespective of the configured failure action.

Dynamic VLAN assignment when RADIUS times out

VLAN assignment for RADIUS timeout actions specified for a client varies. VLAN assignment depends on whether the Flexible authentication-enabled port is configured to be a member of only one untagged VLAN (single untagged mode; default behavior) or whether the port is a member of multiple untagged VLANS (multiple untagged mode).

The RADIUS timeout action can be one of the following:

- Failure
- Success
- Move the client to a critical VLAN

RADIUS timeout actions in single untagged mode

- If a RADIUS timeout action is configured as "failure", the behavior will be the same as that mentioned in Dynamic VLAN assignment in authentication failure scenarios on page 210.
- If a RADIUS 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 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. 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 MAC address is authenticated in the critical VLAN.
  - If the previous sessions are in the auth-default VLAN or RADIUS-assigned VLAN, the MAC address 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 existing MAC sessions are cleared before permitting a new client in the auth-default VLAN or RADIUS-assigned VLAN.

RADIUS timeout actions in 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 will be the same as that mentioned in Dynamic VLAN assignment in authentication failure scenarios on page 210.
- If a RADIUS timeout action is configured as "success", the action is implemented based 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 with the existing dynamic ACL allocation. The VLAN can be either a dynamic untagged or tagged VLAN.
  - In the case of MAC authentication, if the authentication is initiated by a tagged packet, the client is authenticated in the VLAN ID carried by packet's 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.
  - In the case of MAC authentication, if the authentication is initiated by a tagged packet, the client is blocked in the VLAN ID carried by the packet's tag value.

Dynamic VLAN assignment when the client does not respond 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 varies depending on whether the Flexible authentication-enabled port is configured to be a member of only one untagged VLAN (single untagged mode; default behavior) or whether the port is a member of multiple untagged VLANS (multiple untagged mode).

In single untagged mode

- If there is no response from the client for EAP packets and if the guest VLAN is not configured, the behavior will be same as that mentioned in Dynamic VLAN assignment in authentication failure scenarios on page 210.
- 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 which is not a dot1x-capable 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 a guest-vlan and the new client is dot1x-capable, then existing MAC sessions are cleared before permitting the new client in the auth-default VLAN or RADIUS-assigned VLAN.

In multiple untagged mode

- If there is no response from 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 Ruckus 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 device's running-config file. When the **show run** command is issued during a session, dynamically assigned VLANs are not displayed, although they can be displayed with the **show vlan**, **show dot1x sessions**, and **show mac-authentication sessions** commands.

Defining MAC address filters

You can specify MAC addresses that do not have to go through authentication.

These MAC addresses are considered pre-authenticated, and are not subject to authentication. To do this, you can define MAC address filters that specify the MAC addresses to exclude from authentication.

You should use a MAC address filter when the RADIUS server itself is connected to an interface where MAC authentication or 802.1X authentication is enabled. If a MAC address filter is not defined for the MAC address 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 **dot1x auth-filter** command or **mac-authentication auth-filter** command from 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 MAC address filter. The clients authorized in the auth-default VLAN allow both untagged and tagged traffic.

Aging for blocked MAC addresses

Aging for a blocked MAC address occurs in two phases, known as hardware aging and software aging.

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. The software aging period for blocked MAC addresses is configurable, using the **max-sw-age** command (the default is 120 seconds). Once the hardware aging period ends, the software aging period begins. When the software aging period ends, the blocked MAC address ages out, and can be authenticated again if the Brocade device receives traffic from the MAC address. If any traffic is received before the software aging time expires, then both the hardware and software aging timers are reset.

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 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.

Disabling MAC address aging

You can disable aging for all MAC sessions globally or per port to prevent the MAC sessions from being aged out.

You can also disable aging only for permitted (authenticated and restricted) sessions or 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 and software 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 software aging period.

Periodic reauthentication for 802.1X and MAC authenticated clients

NOTE
Reauthentication is supported for restricted and critical VLANs. It is not supported for guest VLANs.

You can configure the device to periodically reauthenticate clients connected to 802.1X-enabled interfaces and MAC authentication-enabled interfaces. When periodic reauthentication is enabled using the **re-authentication** command, the device reauthenticates the clients every 3,600 seconds by default. The reauthentication interval is configurable using the **reauth-period** command. The reauthentication interval configured using the **reauth-period** command takes precedence.

The reauthentication interval configured using the **reauth-period** command can be overwritten for each client by the RADIUS server using the Session-Timeout and Termination-Action attributes.

Dynamic ARP Inspection support for Flexible authentication

Dynamic ARP Inspection (DAI) is inter-operable with Flexible authentication. When Flexible authentication is enabled on a port and the client moves to a RADIUS assigned dynamic VLAN or auth-default VLAN where the DAI is enabled, the DAI feature is executed. Flexible authentication and DAI are also supported in conjunction with dynamic ACLs applied on the port.

NOTE
Flexible authentication interoperability with DAI is supported for both IPv4 and IPv6 networks.

For more information about DAI, refer to the "Dynamic ARP Inspection" section in the "DHCPv4" chapter of the *Brocade FastIron DHCP Configuration Guide*.

DHCP snooping support for Flexible authentication

DHCP snooping is inter-operable with Flexible authentication. When Flexible authentication is enabled on a port and the client moves to a RADIUS assigned dynamic VLAN or auth-default VLAN where the DHCP snooping is enabled, the snooping functions transparently. Flexible authentication and DHCP snooping are also supported in conjunction with dynamic ACLs applied on the port.

NOTE
Both DHCPv4 snooping and DHCPv6 snooping are supported with Flexible authentication.

For more information about DHCP snooping, refer to the "DHCP snooping" and "DHCPv6 snooping" sections in the *Brocade FastIron DHCP Configuration Guide*.

Dynamic IP ACLs and MAC address filters in authentication

NOTE
MAC authentication does not support dynamic assignment of MAC address filters to a port.

After successful authentication, different network policies can be applied to restrict the way the network resources are accessed by the client. The 802.1X authentication and MAC authentication implementations support dynamically applying an IP ACL to a port, based on information received from an authentication server. The 802.1X authentication also supports dynamic assignment of MAC address filters to a port.

When a client or 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 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 device can use information in the attribute to apply an IP ACL or MAC address filter to the authenticated port. This IP ACL or MAC address filter applies to the port for as long as the client is connected to the network. The IP ACL or MAC address filter is removed from the corresponding port when the client logs out, the port goes down, or when the MAC address ages out.

The Ruckus device uses information in the Filter-Id as follows:

- The Filter-Id attribute can specify the number of an existing IP ACL or MAC address filter configured on the Ruckus device. In this case, the IP ACL or MAC address filter with the specified number is applied to the port.
- The attribute can specify actual syntax for a Ruckus IP ACL or MAC address filter, which is then applied to the authenticated 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. The ACL is removed from the corresponding port when the client logs out, the port goes down, or when the MAC address ages out.

Configuration considerations for applying IP ACLs and MAC address filters to 802.1X ports

The following restrictions apply to dynamic IP ACLs or MAC address filters:

- The name in the Filter-Id attribute is case-sensitive.
- You can specify only numbered MAC address filters in the Filter-Id attribute. Named MAC address filters are not supported.
- Dynamically assigned IP ACLs and MAC address filters are subject to the same configuration restrictions as non-dynamically assigned IP ACLs and MAC address filters.
- If dynamically assigned IP ACLs already exist, then MAC address filters cannot be applied dynamically using 802.1X.
- Inbound dynamic IP ACLs are supported. Outbound dynamic ACLs are not supported.
- A maximum of one IP ACL per client can be configured in the inbound direction on an interface.
- 802.1X with a dynamic MAC filter will work for one client at a time on a port. If a second client tries to authenticate with 802.1X and a dynamic MAC filter, the second client will be rejected.
- MAC address filters cannot be configured in the outbound direction on an interface.
- Concurrent operation of MAC address filters and IP ACLs is not supported.
- Static ACLs are not supported on the 802.1X or MAC authentication-enabled port. However, the ACLs can be applied on the VE of the VLAN to which the port belongs. ACLs cannot be applied to the VE interface of the auth-default VLAN.
- Concurrent operation of dynamic IP ACL and static IP ACL is not supported.
- Dynamic IP ACL assignment with 802.1X is not supported in conjunction with any of the following features:
  - Rate limiting
  - Protection against ICMP or TCP Denial of Service (DoS) attacks
  - Policy-based routing

Dynamically applying existing ACLs or MAC address filters

NOTE
MAC authentication does not support dynamic assignment of MAC address filters to a port.

When a port is authenticated, an IP ACL or MAC address filter that exists in the running-config file on the Ruckus device can be dynamically applied to the port. To do this, you configure the Filter-Id (type 11) attribute on the RADIUS server. The Filter-Id attribute specifies the name or number of the Ruckus IP ACL or MAC address filter.

The following is the syntax for configuring the Filter-Id attribute to refer to a Ruckus IP ACL or MAC address filter.

TABLE 19 Syntax for configuring the Filter-Id attribute

Value	Description
ip.*number* .in	Applies the specified numbered ACL to the authenticated port in the inbound direction.
ip.*name* .in	Applies the specified named ACL to the authenticated port in the inbound direction.
mac.*number* .in	Applies the specified numbered MAC address filter to the authenticated port in the inbound direction.

The following table lists examples of values you can assign to the Filter-Id attribute on the RADIUS server to refer to IP ACLs and MAC address filters configured on a Ruckus device.

TABLE 20 Examples of values you can assign to the Filter-Id attribute on the RADIUS server

Possible values for the Filter-Id attribute on the RADIUS server	ACL or MAC address filter 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
mac.2.in	mac filter 2 permit 3333.3333.3333 ffff.ffff.ffff any

NOTE
The dynamic ACL must be an extended ACL. Standard ACLs are not supported.

Strict security mode for dynamic filter assignment

By default, dynamic filter assignment operates in strict security mode. When strict security mode is enabled, authentication for a port fails if the Filter-Id attribute contains invalid information, or if insufficient system resources are available to implement the IP ACLs.

When strict security mode is enabled:

- If the Filter-Id attribute in the Access-Accept message contains a value that does not refer to an existing filter (that is, IP ACL configured on the device), then the client will not be authorized, regardless of any other information in the message (for example, if the Tunnel-Private-Group-ID attribute specifies a VLAN on which to assign the port).
- If the device does not have the system resources available to dynamically apply a filter to a port, then the client will not be authenticated.

NOTE
Also, if authentication for a client fails because the Filter-Id attribute referred to a non-existent filter, or there were insufficient system resources to implement the filter, then a Syslog message is generated.

When strict security mode is disabled:

- If the Filter-Id attribute in the Access-Accept message contains a value that does not refer to an existing filter (that is, a MAC address filter or IP ACL configured on the device), then the client remains authorized and no filter is dynamically applied to it.

By default, strict security mode is enabled for all MAC authentication and 802.1X-enabled interfaces, but you can manually disable or enable it using the **filter-strict-security** command from the authentication configuration mode or using the **authentication filter-strict-security** command from the interface configuration mode.

How Flexible authentication works

Flexible authentication can be configured at the global or interface level.

NOTE
Both 802.1X authentication and MAC authentication methods must be enabled at the global and interface level on the same port to execute Flexible authentication.

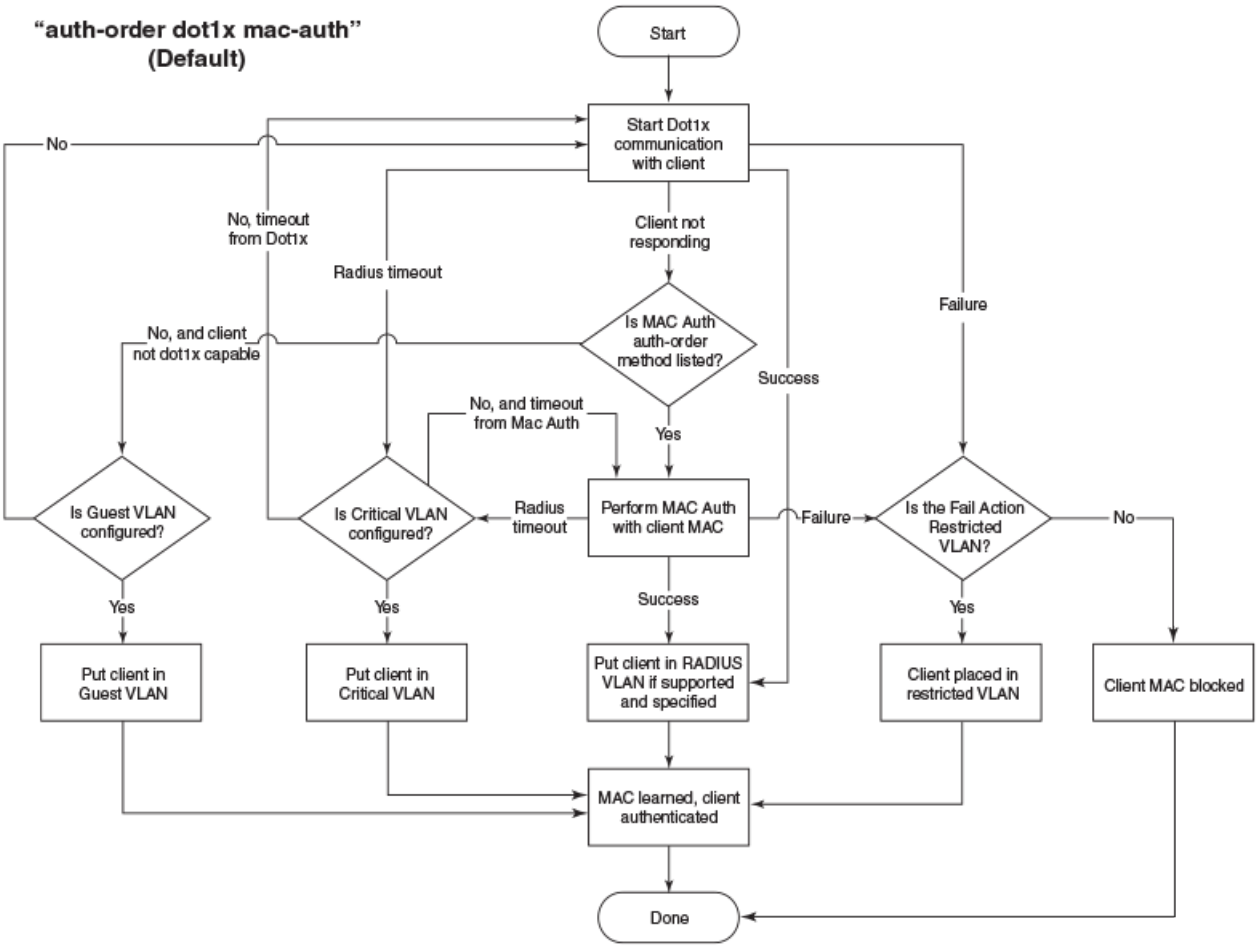
When only 802.1X authentication or MAC authentication is configured, then the configured method is attempted. When authentication fails, the MAC address of the device is blocked (default action) or is moved to a restricted VLAN which can be configured on the switch as the authentication failure action. If authentication succeeds, the client is authenticated and the policies returned by the RADIUS server are applied.

Authentication sequence: 802.1X authentication followed by MAC authentication

When 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 (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, the failure action is carried out and MAC authentication is not attempted. On the other hand, if the client does not respond to dot1x messages, then MAC authentication is attempted. Upon successful MAC authentication, the client is authenticated and the policies returned by the RADIUS server are applied and, on authentication failure, the configured failure action is applied.

FIGURE 6 Authentication sequence: 802.1X authentication followed by MAC authentication



Authentication sequence: MAC authentication followed by 802.1X authentication

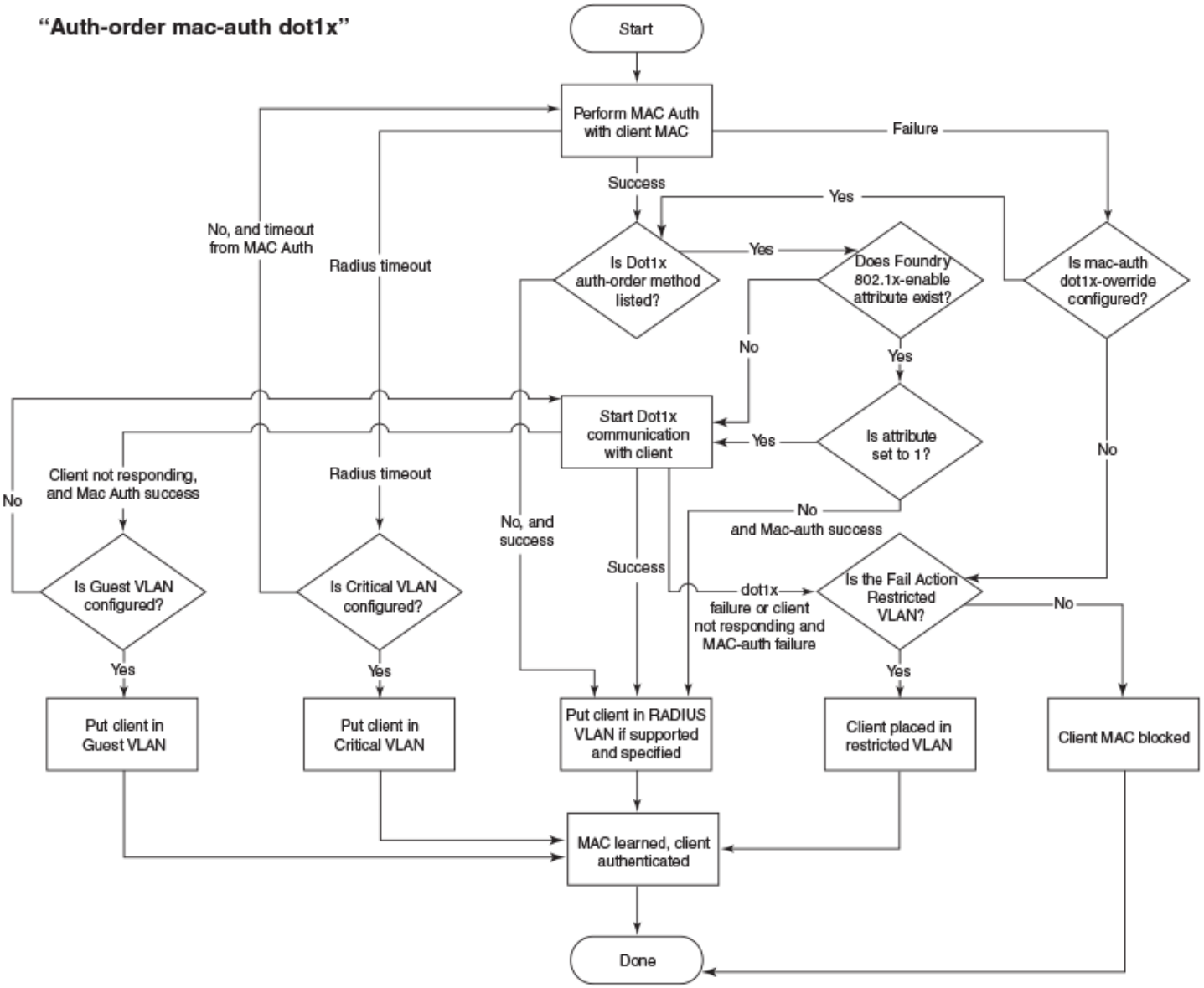
When the authentication sequence is set to perform MAC authentication followed by 802.1X authentication, by default, 802.1X authentication is performed even if MAC authentication is successful (refer to Figure 7). Upon successful 802.1X authentication,

the client is authenticated and the policies returned by the RADIUS server are applied and, on authentication failure, the configured failure action is applied.

The default behavior can be changed by specifying the RADIUS attribute (refer to Table 16 on page 205) to prevent the 802.1X authentication from being performed after successful MAC authentication. In this case, the client is authenticated and the policies returned by the RADIUS server are applied after successful MAC authentication.

When the authentication sequence is set to perform MAC authentication followed by 802.1X authentication, and 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 the failure action is configured as a restricted VLAN. If 802.1X authentication is successful, the policies returned by the RADIUS server are applied to the port.

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:

- If the configured failure action is carried out after the first authentication method in the authentication sequence, the second authentication is not attempted.
- If a dynamic VLAN is not configured or RADIUS does not return any VLAN information in the Access-Accept message, the client is authenticated in the auth-default VLAN.
- If the RADIUS-returned VLAN is not available on the device, the configured failure action is performed.
- When authentication succeeds and RADIUS returns VLAN information, the client is dynamically assigned to the RADIUS-assigned VLANS (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 in authentication success scenarios on page 208.
- When the RADIUS 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. The VLAN can be either a dynamic untagged or tagged VLAN.
- During 802.1X authentication, when a RADIUS-returned Layer 2 MAC filter does not exist on the switch, the client authentication fails by default.
- During 802.1X authentication, when a RADIUS-returned MAC filter does not exist on the switch and the **no filter-strict-security** command is configured, the client is authenticated.

Configuration considerations and guidelines for Flexible authentication

- Before configuring Flexible authentication, communication between the devices and the authentication server must be established.
- You cannot configure dual-mode on a Flexible authentication-enabled port. By default, a port supports untagged traffic. It can also support tagged traffic if it is a tagged member of a VLAN.
- You cannot enable Flexible authentication on ports that have any of the following features enabled:
  - Link aggregation
  - Metro Ring Protocol (MRP)
  - Mirror port
  - LAG port
  - Unidirectional Link Detection (UDLD)
- Incoming traffic on unauthenticated ports is blocked by Brocade 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.
- When the authentication sequence is set to perform 802.1X authentication followed by MAC authentication, and if the client is 802.1X capable, MAC authentication is not performed.
- When the authentication sequence is set to perform MAC authentication followed by 802.1X authentication, and if the client is 802.1X capable, 802.1X authentication is performed even if MAC authentication is successful.

- If Web authentication is enabled on the RADIUS-assigned VLAN where the client is placed after successful 802.1X authentication or MAC authentication, Web authentication is also performed.
- If Web authentication is enabled on restricted VLAN, critical VLAN, or guest VLAN that are configured to associate the clients in various failure and timeout scenarios, the device uses Web authentication as a fallback. Web authentication can be enabled on any 2 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 fixed hardware aging period (70 seconds), plus a configurable software aging period. You can optionally change the software aging period for the session or disable aging altogether. After the denied client session is aged out, traffic from that client is no longer blocked, and the client can be re-authenticated.

Configuration considerations and guidelines for dynamic ACL and dynamic MAC filters

- Dynamic MAC address filters with MAC authentication are not supported.
- In the Layer 2 switch code, dynamic IP ACLs are not supported when ACL-per-port-per VLAN is enabled on a global basis.
- The dynamic ACL must be an extended ACL. Standard ACLs are not supported.
- MAC authentication and 802.1X authentication can be used together on the same port.
- Dynamically assigned IP ACLs are subject to the same configuration restrictions as non-dynamically assigned IP ACLs.
- Dynamic ACL filters are supported only for the inbound direction. Dynamic outbound ACL filters are not supported.
- Dynamic ACL assignment with MAC authentication is not supported in conjunction with any of the following features:
  - Rate limiting
  - Protection against ICMP or TCP Denial of Service (DoS) attacks
  - Policy-based routing

Support for authenticating multiple MAC sessions 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 and can be changed using the **authentication max-sessions** command. The maximum number of authenticated MAC sessions on an interface depends on the Brocade device and dynamic ACL assignments. If RADIUS assigns dynamic ACL to at least one client on the interface, the maximum number of MAC sessions that can be authenticated is limited to 32 in all Brocade devices.

If 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 Brocade device as specified in Table 21. 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 21 Maximum number of authenticated MAC sessions per port on various platforms

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 7750	1024	32
ICX 7450	1024	32
ICX 7250	1024	32

The system limit for authenticated MAC sessions also varies and depends on the Brocade device and dynamic ACL assignments.

TABLE 22 Maximum number of authenticated MAC sessions per system (standalone or stack) on various platforms

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 7750	1536	512
ICX 7450	1536	512
ICX 7250	1536	512

Support for IP source guard protection

The Ruckus 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 from the 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 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 comprises DHCP Snooping and Static ARP Inspection entries.

The IP Source Guard ACL entry is not written to the running configuration file. However, you can view the configuration using the **show mac-authentication sessions** command or the **show dot1x sessions** command at the global level or for a specific interface.

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 doesn't get affected if the DHCP Secure table is updated after the session is authenticated and while the session is still active.

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 the "IP Source Guard" section in the "DHCPv4" chapter of the *Brocade FastIron DHCP Configuration Guide*.

Denial of Service protection support

A Denial of Service (DoS) attack can occur against the Brocade device where 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 from the interface configuration mode. The Brocade 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.

802.1X authentication

Ruckus FastIron devices support the IEEE 802.1X standard for authenticating devices attached to LAN ports. Using 802.1X, you can configure a FastIron 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 Ruckus 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 Ruckus 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

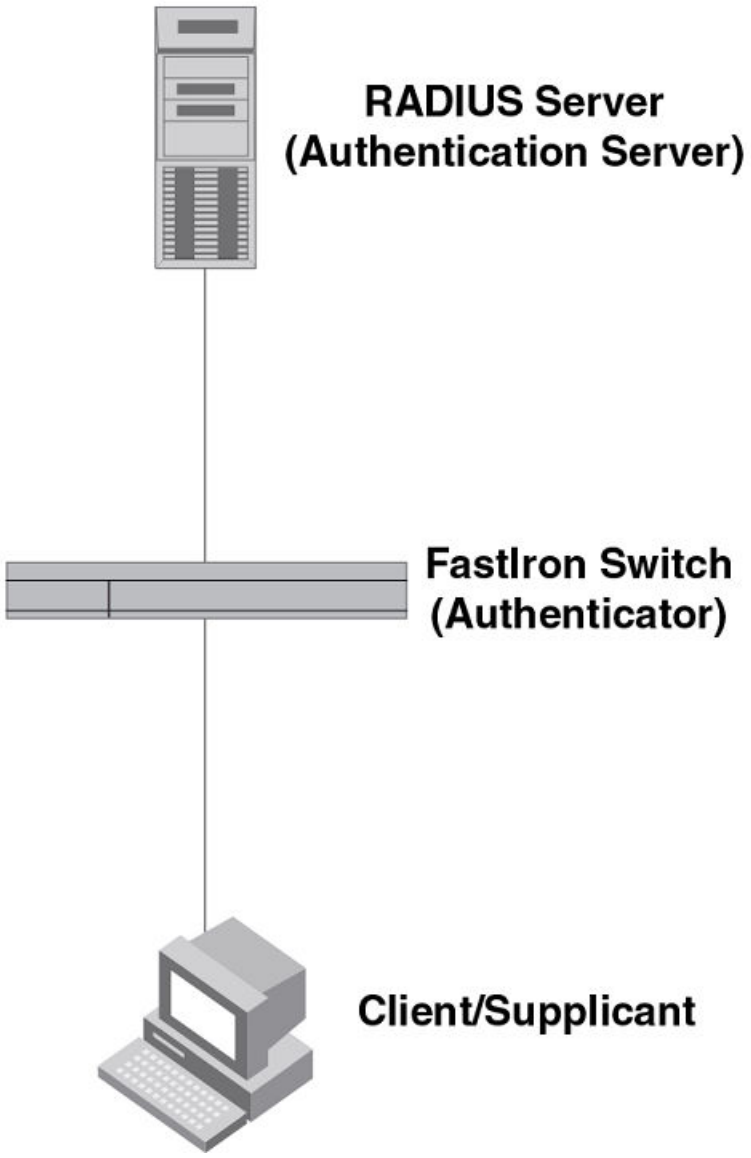
Device roles in an 802.1X configuration

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 Ruckus 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 XP operating system). Clients can either be directly connected to a port on the authenticator, or can be connected by way of a hub.

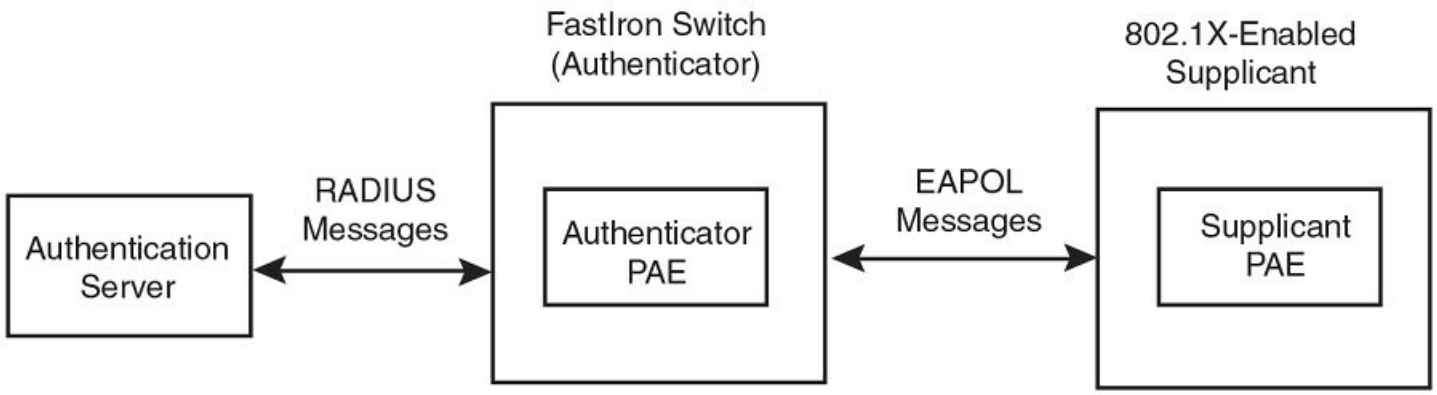
Authentication server: The device that validates the client and specifies whether or not the client may access services on the device. Ruckus supports authentication servers running RADIUS.

Communication between the devices

For communication between the devices, 802.1X uses the Extensible Authentication Protocol (EAP), defined in RFC 2284. The 802.1X standard specifies a method for encapsulating EAP messages so that they can be carried over a LAN. This encapsulated form of EAP is known as EAP over LAN (EAPOL). The standard also specifies a means of transferring the EAPOL information between the client/supplicant, authenticator, and authentication server.

EAPOL messages are passed between the Port Access Entity (PAE) on the supplicant and the authenticator. The following figure shows the relationship between the authenticator PAE and the supplicant PAE.

FIGURE 9 Authenticator PAE and supplicant PAE



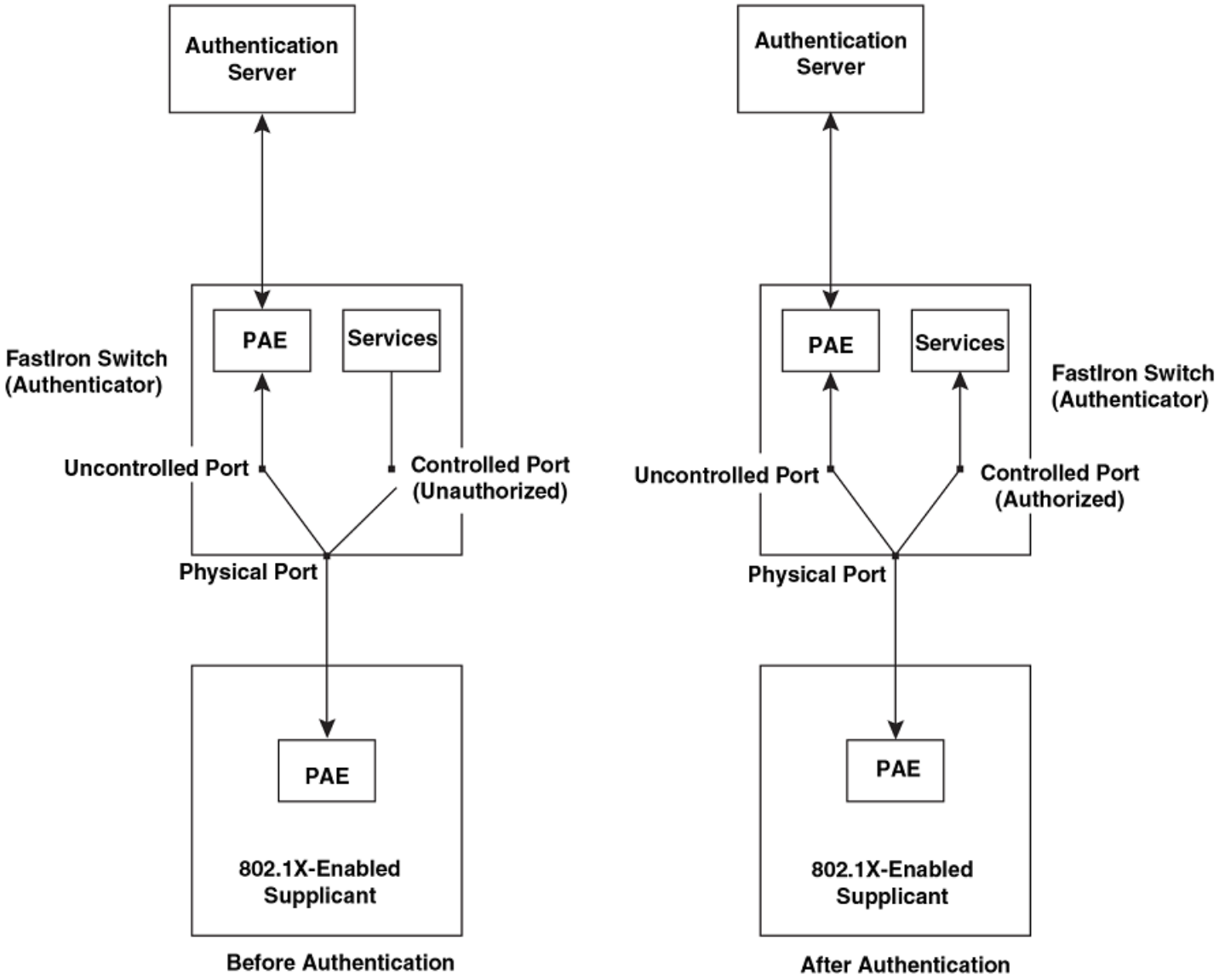
Authenticator PAE: The authenticator PAE communicates with the supplicant PAE, receiving identifying information from the supplicant. Acting as a RADIUS client, the authenticator PAE passes the supplicant information to the authentication server, which decides whether the supplicant can gain access to 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, and traffic from the client can flow through the port normally.

By default, all controlled ports on the Ruckus device are placed in the authorized state, allowing all traffic. When authentication is activated on an 802.1X-enabled interface, the interface controlled port is placed initially in the unauthorized state. When a client connected to the port is successfully authenticated, the controlled port is then placed in the authorized state until the client logs off.

Setting the port control

To activate authentication on an 802.1X-enabled interface, you specify the kind of port control to be used on the interface. An interface used with 802.1X authentication has two virtual access points: a controlled port and an uncontrolled port:

- The controlled port can be in either the authorized or unauthorized state. In the authorized state, it allows normal traffic to pass between the client and the authenticator. In the unauthorized state, no traffic is allowed to pass.
- The uncontrolled port allows only EAPOL traffic between the client and the authentication server.

The port control type can be one of the following:

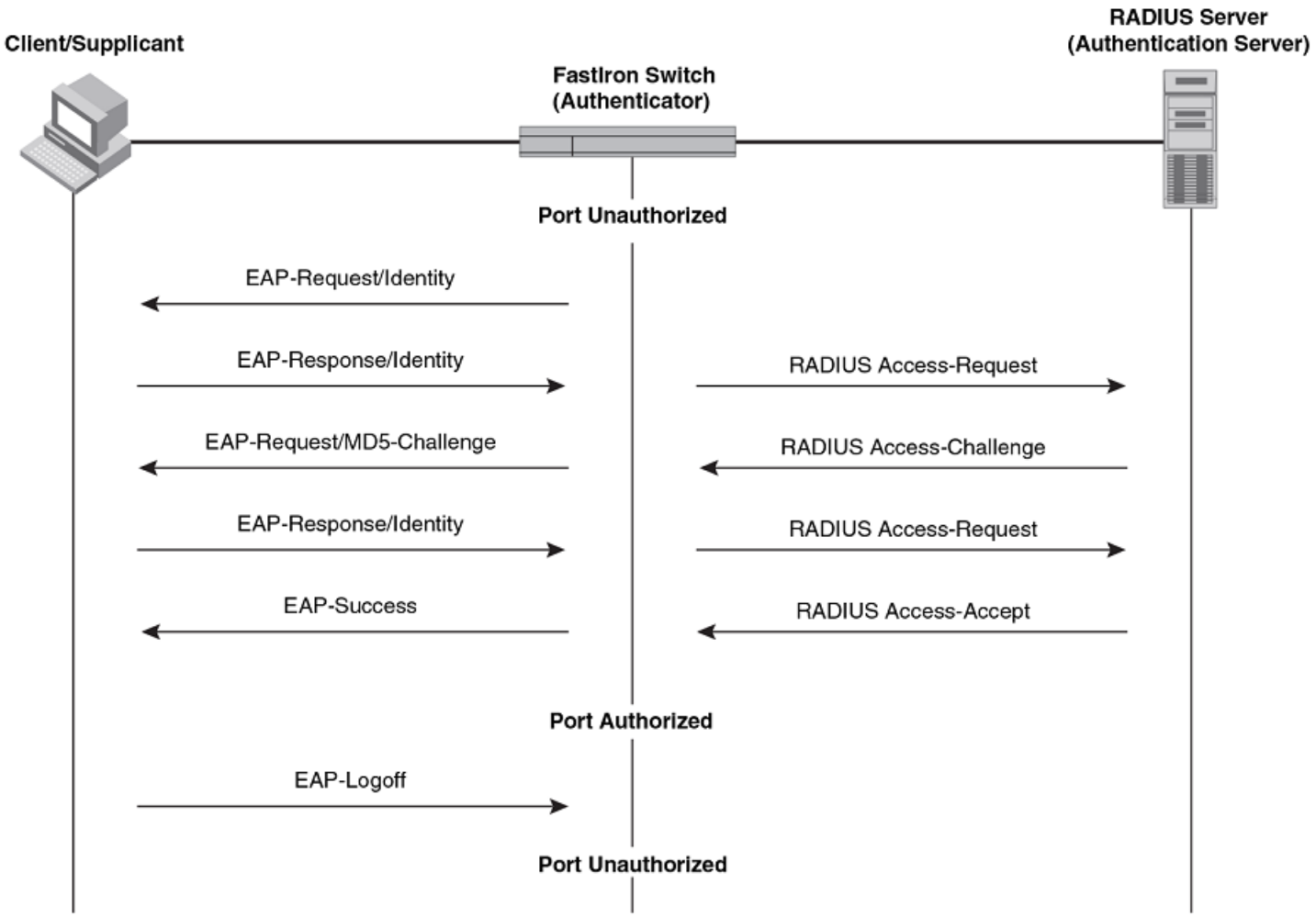
- force-authorized: The controlled port is placed unconditionally in the authorized state, allowing all traffic. This is the default state for ports on the Ruckus device.
- force-unauthorized: The controlled port is placed unconditionally in the unauthorized state.
- auto: The controlled port is unauthorized until authentication takes place between the client and authentication server. Once the client passes authentication, the port becomes authorized. This activates authentication on an 802.1X-enabled interface.

NOTE
Before activating the authentication using the **dot1x port-control auto** command on an untagged port, you must remove the configured static ACL, if any, from the port.

Message exchange during authentication

The following figure illustrates a sample exchange of messages between an 802.1X-enabled client, a FastIron switch acting as 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 FastIron switch) 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 Ruckus 802.1X implementation supports dynamic VLAN assignment. If one of the attributes in the Access-Accept message sent by the RADIUS server specifies a VLAN identifier, and this VLAN is available on the Ruckus device, 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. Refer to Flexible authentication with dynamic VLAN assignment on page 207 for more information.

If a client does not support 802.1X, authentication cannot take place. The Ruckus 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 Ruckus device. When the device does not respond, the client considers the port to be authorized, and starts sending normal traffic.

Ruckus 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 client and 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 users' 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, which does not need to understand the specific EAP authentication protocols.

  Unlike EAP-TTLS, PEAP does not natively support username and password to authenticate clients against an existing user database such as LDAP. PEAP secures the transmission between the client and 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 *Brocade FastIron Layer 3 Routing Configuration Guide*.

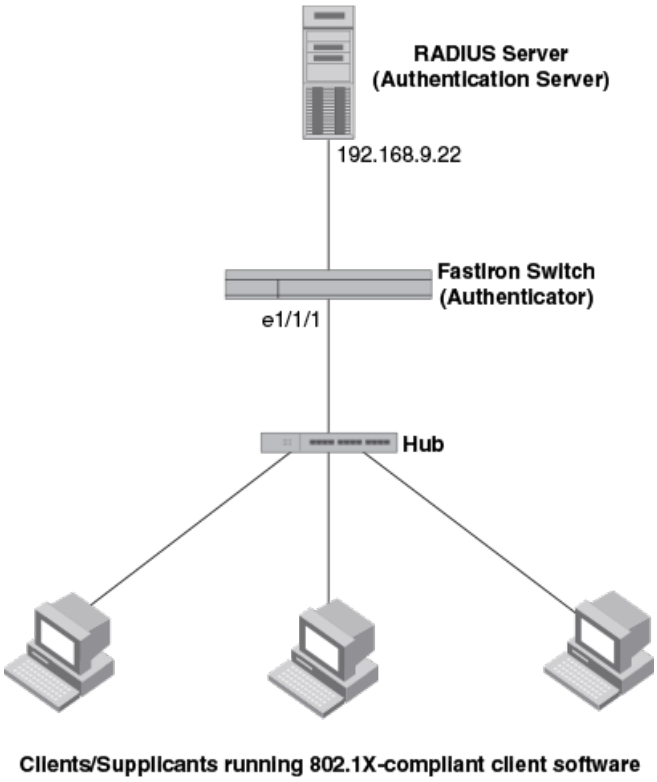
EAP pass-through support

EAP pass-through is supported on FastIron 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. EAP pass-through, which can be configured using the **pass-through** command, also allows protocol packets such as Link Layer Discovery Protocol (LLDP), Foundry Discovery Protocol (FDP), and Cisco Discovery Protocol (CDP) to pass through the ports enabled for Flexible authentication.

Authenticating multiple hosts connected to the same port

Ruckus 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 802.1X-enabled port.

FIGURE 12 Multiple hosts connected to a single 802.1X-enabled port



If there are multiple hosts connected to a single 802.1X-enabled port, the Ruckus 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. You can optionally configure the Ruckus device to assign the port to a "restricted" VLAN if authentication of the client is unsuccessful.

How 802.1X host authentication works for multiple clients

When multiple hosts are connected to a single 802.1X-enabled port on a Ruckus device, 802.1X authentication is performed in the following way.

1. One of the 802.1X-enabled clients attempts to log in to a network in which a Ruckus device serves as an authenticator.
2. The Ruckus device creates an internal session (called a dot1x-mac-session) for the client. A dot1x-mac-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) need to be authenticated from each client.
3. The Ruckus device performs 802.1X authentication for the client. Messages are exchanged between the Ruckus device and the client, and between the device and the authentication server (RADIUS server). The result of this process is that the client is either successfully authenticated or not authenticated, based on the username and password supplied by the client.
4. If the client is successfully authenticated, the client dot1x-mac-session is set to "access-is-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 taken. The authentication-failure action can be either to drop traffic from the client, or to place the port in a restricted VLAN:
   - If the authentication-failure action is to drop traffic from the client, then the client dot1x-mac-session is set to "access-denied", causing 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 dot1x-session is set to "access-restricted". The port is moved to the specified restricted VLAN, and traffic from the client is forwarded normally.
6. When the client disconnects from the network, the Ruckus device deletes the client dot1x-mac-session. This does not affect the dot1x-mac-session or authentication status (if any) of the other hosts connected on the port.
7. If a client has been denied access to the network (that is, the client dot1x-mac-session is set to "access-denied"), then you can re-authenticate the client manually by disconnecting the client from the network, or by using the **clear dot1x sessions** command.

   NOTE
   Dynamic IP ACL and MAC address filter assignment is supported in an 802.1X multiple-host configuration. Refer to Dynamic IP ACLs and MAC address filters in authentication on page 214.

802.1X accounting

When 802.1X authentication is enabled on the Ruckus device, you can enable 802.1X accounting. This feature enables the Ruckus device to log information on the RADIUS server about authenticated 802.1X clients. The information logged on the RADIUS server includes the 802.1X client session ID, MAC address, and authenticating physical port number.

802.1X accounting works as follows.

1. A RADIUS server successfully authenticates an 802.1X client.
2. If 802.1X accounting is enabled, the Ruckus 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. When the session is concluded, the Ruckus device sends an Accounting Stop packet to the RADIUS server, indicating the end of the session.
6. The RADIUS server acknowledges the Accounting Stop packet.

   To enable 802.1X accounting, refer to 802.1X accounting configuration on page 79.

MAC authentication

MAC authentication is a way to configure a Ruckus device to forward or block traffic from a MAC address based on information received from a RADIUS server.

MAC authentication is a mechanism by which incoming traffic originating from a specific MAC address is switched or forwarded by the device only if the source MAC address is successfully authenticated by a RADIUS server. 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. If RADIUS authentication for the MAC address is successful, traffic from the MAC address is forwarded in hardware.

If the RADIUS server cannot validate the user's MAC address, then it is considered an authentication failure, and a specified authentication-failure action can be taken. 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.

How MAC authentication works

MAC authentication communicates with the RADIUS server to authenticate a newly found MAC address. The Ruckus 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 session times out, and the device retries the request up to three times. If no response is received, the next RADIUS server is chosen, and the request is sent for authentication.

The RADIUS server is configured with the usernames and passwords of authenticated users. For MAC authentication, the username and password is the MAC address itself; that is, the device uses the MAC address for both the username and the password in the request sent to the RADIUS server. For example, given a MAC address of 0000000feaa1, the user's file on the RADIUS server would be configured with the username and password both set to 0000000feaa1. When traffic from this MAC address is encountered on a MAC-authentication-enabled interface, the device sends the RADIUS server an Access-Request message with 0000000feaa1 as both the username and password. The format of the MAC address sent to the RADIUS server can be configured using the **mac-authentication password-format** command. You can also specify a password instead of the MAC address for authentication using the **mac-authentication password-override** command.

The request for authentication from the RADIUS server is successful only if the username and password provided in the request matches an entry in the user database on the RADIUS server. When this happens, the RADIUS server returns an Access-Accept message back to the Ruckus device. When the RADIUS server returns an Access-Accept message for a MAC address, that MAC address is considered authenticated, and traffic from the MAC address is forwarded normally by the Ruckus device.

SNMP traps for MAC authentication

You can enable and disable SNMP traps for MAC authentication using the **snmp-server enable traps mac-authentication** command. SNMP traps are enabled by default.

Format of the MAC addresses sent to the RADIUS server

The MAC address of the device is used as the username and password for authentication.

When MAC authentication is configured, the Brocade device authenticates MAC addresses by sending username and password information to a RADIUS server. The device uses the MAC address for both the username and the password in the request sent to the RADIUS server. You can configure the format in which MAC address is sent to the RADIUS server for authentication using the **mac-authentication password-format** command. For ease of configuration and depending on the RADIUS server you use, you can opt to send the password in uppercase. The **lowercase** option is used by default.

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.

Configuration prerequisites

Before configuring Flexible authentication, communication between the devices and the authentication server must be established. The following steps involve the configurations that are required before configuring Flexible authentication.

- Configure the device interaction with the authentication server by configuring an authentication method list for 802.1X and specifying RADIUS as an authentication method. For more information, refer to AAA operations for RADIUS on page 61.

```
device(config)# aaa authentication dot1x default radius
```

- Configure the RADIUS server to authenticate access to the Brocade device. For more information, refer to AAA operations for RADIUS on page 61.

```
device(config)# radius-server host 10.20.64.208 auth-port 1812 acct-port 1813 default key secretkey
```

- After successful authentication, the client is moved to the RADIUS-assigned VLAN. For RADIUS-based VLAN assignments, VLANs must be preconfigured on the switch.

  Configure a VLAN as the auth-default VLAN to enable authentication. When any port is enabled for 802.1X authentication 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 authentication failure and timeout scenarios.

```
device(config)# vlan 2 name auth-default-vlan
```

- After a successful authentication, user access can be limited by using ACLs. ACLs must be preconfigured on the switch and the RADIUS server can return the ACL ID or name. If the ACL matches with 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 “any” because the Brocade switch dynamically learns the IP addresses of the clients (source). The destination network is user-configurable.

  For more information on ACL configuration, refer to IP ACLs on page 111. For more information about dynamic ACL assignment, refer to Dynamic IP ACLs and MAC address filters in authentication on page 214.

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.

```
device(config-authen)# auth-order mac-auth dot1x
```

   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.

4. Enter the **auth-default-vlan** command to configure the authentication default VLAN (auth-default VLAN).

```
device(config-authen)# auth-default-vlan 2
```

   The auth-default VLAN must be configured to enable Flexible authentication before enabling 802.1X authentication or MAC authentication.

   The client's MAC address is moved to the auth-default VLAN as a default MAC VLAN member upon enabling 802.1X authentication or MAC authentication. 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".

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.

   NOTE
   A VLAN configured as a restricted VLAN must be a valid VLAN configured on the device.

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
```

   NOTE
   A restricted VLAN must be configured before setting the authentication failure action as the restricted VLAN.

   When the authentication failure action is configured as a restricted VLAN, the client is moved to the restricted VLAN when authentication fails. If the authentication failure action is not configured, the client's MAC address is blocked in the hardware (default action) when the authentication fails.

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
```

   When a critical VLAN is configured and the authentication timeout action is specified as a critical VLAN under the port using the **authentication timeout-action critical-vlan** command at the interface level and, if RADIUS timeout occurs, the client is moved to the critical VLAN and any access policies applied to the critical VLAN are applied to the client.

   NOTE
   The VLAN configured as a critical VLAN must be a valid VLAN configured on the device.

8. (Optional) Enter the **auth-vlan-mode** command to enable multiple untagged mode, which allows Flexible authentication-enabled ports to be members of multiple untagged VLANs.

```
device(config-authen)# auth-vlan-mode multiple-untagged
```

   By default, a Flexible authentication-enabled port can be a member of only one untagged VLAN, and other clients that are authenticated with different dynamic untagged VLANs are blocked.

9. (Optional) Enter the **disable-aging** command to prevent the permitted or denied MAC sessions from being aged out.

```
device(config-authen)# disable-aging permitted-mac-only
```

```
device(config-authen)# disable-aging denied-mac-only
```

   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. By default, **max-sw-age** is set to 120 seconds. After the software aging period ends, the client session ages out and can be authenticated again if the Ruckus ICX device receives traffic from the MAC address. 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 Ruckus ICX device receives traffic from the MAC address.

10. (Optional) Enter the **max-hw-age** command to configure the hardware aging period for denied MAC addresses.

```
device(config-authen)# max-hw-age 160
```

11. (Optional) Enter the **max-sw-age** command to configure the software aging period.

```
device(config-authen)# max-sw-age 160
```

12. (Optional) Enter the **pass-through** command to configure pass-through support, which allows certain protocol packets to pass through ports that are enabled for Flexible authentication.

```
device(config-authen)# pass-through lldp
```

   You can enable LLDP, FDP, and CDP packets to pass through the port.

13. (Optional) Enter the **re-authentication** command to configure the device to periodically reauthenticate the clients connected to 802.1X authentication-enabled and MAC authentication-enabled interfaces.

```
device(config-authen)# re-authentication
```

   NOTE
   Reauthentication is supported for restricted and critical VLANs. It is not supported for guest VLANs.

   When the periodic reauthentication is enabled, the device reauthenticates clients every 3,600 seconds by default. The reauthentication interval configured using the **reauth-period** command takes precedence.

14. (Optional) Enter the **reauth-period** command to configure the interval at which clients connected to 802.1X authentication-enabled and MAC authentication-enabled ports are reauthenticated.

```
device(config-authen)# reauth-period 2000
```

Configuring Flexible authentication on an interface

The following steps configure Flexible authentication at the interface level.

NOTE
Configuration steps 3 through 8 executed at the interface level override the values configured at the global level. The global configurations will still be applicable to other ports that do not have a per-port configuration. Configuration steps 9 through 13 can be performed only at the interface level.

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 method as MAC authentication followed by 802.1X authentication if required.

```
device(config-if-e1000-1/1/1)# authentication auth-order mac-auth dot1x
```

   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.

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 3
```

   The client's MAC address is moved to the auth-default VLAN as a default MAC VLAN member upon enabling 802.1X authentication or MAC authentication. The client is authenticated 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". However, the client is authenticated in the auth-default VLAN upon RADIUS timeout with the success action only if the RADIUS timeout occurs during the first authentication attempt. If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN.

5. (Optional) Enter the **authentication auth-vlan-mode** command to enable multiple-untagged mode on a specific Flexible authentication-enabled port and allow it to be member of multiple untagged VLANs.

```
device(config-if-e1000-1/1/1)# authentication auth-vlan-mode multiple-untagged
```

   By default, a Flexible authentication-enabled port can be a member of only one untagged VLAN, and other clients that are authenticated with different dynamic untagged VLANs are blocked.

6. (Optional) Enter the **authentication fail-action** command to move the client to the restricted VLAN after a MAC authentication or 802.1X authentication failure at the interface.

```
device(config-if-e1000-1/1/1)# authentication fail-action restricted-vlan 2
```

   When an authentication failure action is configured as a restricted VLAN and if authentication fails, the client is moved to the restricted VLAN. If the authentication failure action is not configured, the client's MAC address is blocked in the hardware (default action) when authentication fails.

   The restricted VLAN specified at the interface level overrides the restricted VLAN configured using the **restricted-vlan** command at the global level.

7. (Optional) Enter the **no authentication filter-strict-security** command to authenticate the client on a specific interface even if the Filter-Id attribute returned by RADIUS contains invalid information, or if insufficient system resources are available to implement the IP ACLs.

```
device(config-if-e1000-1/1/1)# no authentication filter-strict-security enable
```

   By default, strict security mode is enabled.

8. (Optional) Enter the **authentication disable-aging** 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
device(config-if-e1000-1/1/1)# authentication disable-aging denied-mac-only
```

   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, known as MAC aging interval configured using the **mac-age-time** command and software aging. 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 Brocade device receives traffic from the MAC address.

9. Configure the authentication timeout actions to specify the action for the RADIUS server if an authentication timeout occurs using one of the following methods:
   - Enter the **authentication timeout-action success** command to consider the client as authenticated after RADIUS timeout.

```
device(config-if-e1000-1/1/1)# authentication timeout-action success
```

     When the RADIUS 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 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. The VLAN can be either a dynamic untagged or tagged VLAN.
   - Enter the **authentication timeout-action failure** command to follow the configured failure action. If the failure action is not configured, the client's MAC address is blocked in the hardware.

```
device(config-if-e1000-1/1/1)# authentication timeout-action failure
```

     If the authentication failure action is configured as a restricted VLAN using the **authentication fail-action** command, the client is placed in the restricted VLAN. A restricted VLAN must be configured using the **restricted-vlan** command at the global level or the **authentication fail-action restricted-vlan** command at the interface level if the authentication failure action is configured.
   - Enter the **authentication timeout-action critical-vlan** command to move the client to the specified critical VLAN.

```
device(config-if-e1000-1/1/1)# authentication timeout-action critical-vlan 3
```

     The critical VLAN specified at the interface level overrides the critical VLAN configured using the **critical-vlan** command at the global level.

10. (Optional) Enter the **authentication reauth-timeout** command to set the time to wait before reauthenticating a client after a timeout action (critical-vlan) is applied.

```
device(config-if-e1000-1/1/1)# authentication reauth-timeout 100
```

11. (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
```

    The maximum number of authenticated MAC sessions on an interface depends on the Brocade device and the dynamic ACL assignments.

12. (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
```

    You can also configure the Brocade device to limit the rate of authentication attempts sent to the RADIUS server.

13. (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
```

Enabling 802.1X authentication

The following steps are for enabling and activating 802.1X authentication and also 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** *stack/slot/pot* } command to enable 802.1X authentication on all interfaces or a specific interface.

```
device(config-authen)# dot1x enable all
```

    NOTE
    Port control must be configured to activate authentication on an 802.1X-enabled interface using the **dot1x port-control auto** command from interface configuration mode.

    NOTE
    Before activating the authentication using the **dot1x port-control auto** command on an untagged port, you must remove configured static ACLs, if any, from the port.

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 authentication server.

```
device(config-if-e1000-1/1/1)# dot1x port-control auto
```

   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
```

   If there is no response from the dot1x client for EAP packets and if a guest VLAN is not configured, authentication is considered as failed and the configured failure action is performed.

7. (Optional) Configure the timeout parameters that determine the time interval for client re-authentication and EAP retransmissions using the following commands:
   - Enter the **dot1x timeout quiet-period** command to configure the amount of time the 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 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 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, client reauthentication and retransmission of EAP-Request/Identity frames and EAP-Request/Identity frames is performed.

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 (or the amount of time specified with the **auth-max** command), 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 EAP Response/Identity frame is not received from the client.

```
device(config-authen)# dot1x max-req 3
```

10. (Optional) Enter the **dot1x auth-filter** command to apply the specified filter on the interface and the MAC addresses defined in the filter (MAC filter) do not have to go through authentication.

    ```
    device(config-if-e1000-1/1/1)# dot1x auth-filter 2 4
    ```

    The source MAC addresses defined using the **mac filter** command are considered pre-authenticated, and are not subject to 802.1X authentication. A client can be authenticated in an untagged VLAN or tagged VLAN using the MAC address filter for 802.1X authentication. 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 MAC authentication

The following steps enable MAC authentication and also 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.

    ```
    device(config-authen)# mac-authentication password-format xx-xx-xx-xx-xx-xx upper-case
    ```

    By default, the MAC address is sent to the RADIUS server in the xxxxxxxxxxxx format in lower case.

6. (Optional) Enter the **mac-authentication password-override** command to specify a user-defined password instead of the MAC address for MAC authentication.

    ```
    device(config-authen)# mac-authentication password-override ts54fs
    ```

    The password can contain up to 32 alphanumeric characters, but must not include blank spaces.

7. (Optional) Enter the **mac-authentication dot1x-override** command to configure the device to perform 802.1X authentication after MAC authentication.

    ```
    device(config-authen)# mac-authentication dot1x-override
    ```

    This command is applicable only when the authentication sequence is configured as MAC authentication followed by 802.1X authentication.

    If the **mac-authentication dot1x-override** command is configured, the clients that failed MAC authentication undergo 802.1X authentication if the failure action is configured as a restricted VLAN.

8. (Optional) Enter the **mac-authentication auth-filter** command to apply the specified filter on the interface, and the MAC addresses defined in the filter (MAC filter) do not have to go through authentication.

```
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# mac-authentication auth-filter 1 2
```

The source MAC addresses defined using the **mac filter** command are considered pre-authenticated, and are not subject to MAC authentication. A client can be authenticated in an untagged VLAN or tagged VLAN using the MAC address filter for MAC authentication. 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.

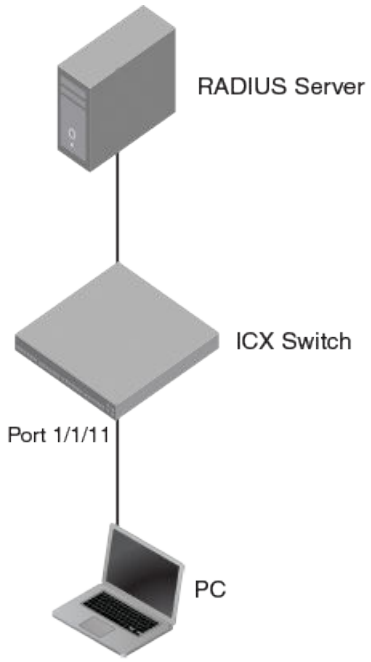
Configuration examples

A number of use cases illustrate the various configurations available in flexible authentication.

Use case 1: 802.1X authentication with dynamic VLAN assignment

This use case shows the configuration required on a Brocade switch to authenticate an 802.1X client and assign the client to a VLAN dynamically. In the following example, after authentication, the PC will be placed in VLAN 200.

FIGURE 13 802.1X authentication with dynamic VLAN assignment



RADIUS configuration

Create a user profile on the RADIUS server and configure the attributes in the following table.

TABLE 23 RADIUS attributes for PC user

Attribute	Value
Tunnel-Medium-Type	802
Tunnel-Pvt-Group-ID	200
Tunnel-Type	VLAN

Brocade switch configuration

1. Specify RADIUS as an authentication server. The following command configures the switch to use the configured RADIUS server to authenticate 802.1X authentication or MAC authentication clients.

```
device(config)# aaa authentication dot1x default radius
```

2. Configure a RADIUS server. In the following example, the RADIUS server IP address is 10.20.64.208 and the shared key is "secret". The shared key should match the key given during client configuration on the RADIUS server. UDP port 1812 is used for RADIUS authentication messages and UDP port 1813 is used for RADIUS accounting messages.

```
device(config)# radius-server host 10.20.64.208 auth-port 1812 acct-port 1813 default key secret
```

3. Create a VLAN to be used as the auth-default VLAN. This VLAN must be configured to enable authentication. When any port is enabled for 802.1X authentication or MAC authentication, the port is moved into this VLAN by default as a MAC VLAN member. Sometimes the RADIUS server may authenticate the client but not return VLAN information on where the client should be placed. The auth-default VLAN is used in this scenario.

```
device(config)# vlan 2 name auth-default-vlan
device(config-vlan-2)# exit
```

4. Create a VLAN that will be used as a restricted VLAN. This VLAN must be active in the Brocade switch. A VLAN is active when it has at least one untagged or tagged member port. In the following example, VLAN 100 is made active by adding the unused port 2/1/13 as an untagged member.

```
device(config)# vlan 100
device(config-vlan-100)# untagged ethernet 2/1/13
device(config-vlan-100)# exit
```

5. Create the VLANs that will be assigned to clients by RADIUS. RADIUS will return VLAN 200 for the PC. This VLAN must be active in the Brocade switch. A VLAN is active when it has at least one untagged or tagged member port. In this example, VLAN 200 is made active by adding the unused port 2/1/12 as an untagged member.

```
device(config)# vlan 200
device(config-vlan-200)# untagged ethernet 2/1/12
device(config-vlan-200)# exit
```

6. Specify which VLAN ID to use as the auth-default VLAN under authentication mode. Refer to step 3 for the use of the auth-default VLAN.

```
device(config)# authentication
device(config-authen)# auth-default-vlan 2
```

7. To configure the authentication failure action as a restricted VLAN, specify a VLAN ID to be used as the restricted VLAN, and then configure the authentication failure action as the restricted VLAN. In this example, VLAN 100 is configured to be used as the restricted VLAN.

```
device(config-authen)# restricted-vlan 100
device(config-authen)# auth-fail-action restricted-vlan
```

8. Enable 802.1X on the switch under authentication mode and enable 802.1X on port 1/1/11. Configure the port control mode as **auto** in interface configuration mode. The **auto** mode enables the 802.1X authentication on the interface.

```
device(config-authen)# dot1x enable
device(config-authen)# dot1x enable ethernet 1/1/11
device(config-authen)# exit
device(config)# interface ethernet 1/1/11
device(config-if-e1000-1/1/11)# dot1x port-control auto
device(config-if-e1000-1/1/11)# exit
```

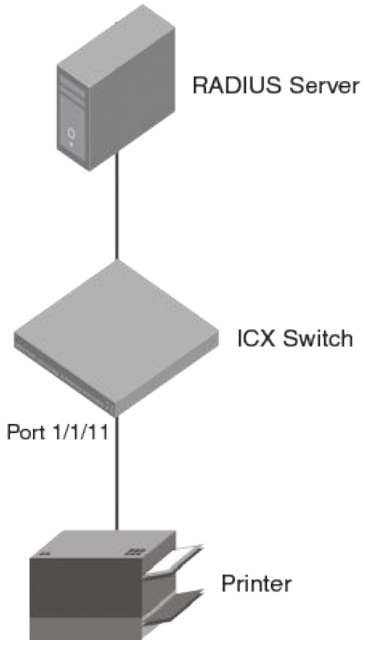
9. To verify the authentication-related configuration on the switch, use the **show running-configuration | begin authentication** command. Authentication-related configurations are stored under the keyword "authentication".

```
device# show running-configuration | begin authentication
authentication
 auth-default-vlan 2
 restricted-vlan 100
 auth-fail-action restricted-vlan
 dot1x enable
 dot1x enable ethe 1/1/11
!
```

Use case 2: MAC authentication with dynamic VLAN assignment

This use case shows the configuration required on a Brocade switch to authenticate a non-802.1X-capable client by way of MAC authentication and assign the client to a VLAN dynamically. In the following example, after authentication, the printer will be placed in VLAN 200.

FIGURE 14 MAC authentication with dynamic VLAN assignment



RADIUS configuration

Create a device profile for the printer's MAC address on the RADIUS server and configure following attributes.

TABLE 24 RADIUS attributes for printer profile

Attribute	Value
Tunnel-Medium-Type	802
Tunnel-Pvt-Group-ID	200
Tunnel-Type	VLAN

Brocade switch configuration

1. Specify RADIUS as an authentication server. The following command configures the switch to use the configured RADIUS server to authenticate 802.1X authentication or MAC authentication clients.

```
device(config)# aaa authentication dot1x default radius
```

2. Configure a RADIUS server. In the following example, the RADIUS server IP address is 10.20.64.208 and the shared key is "secret". The shared key should match the key given during client configuration on the RADIUS server. UDP port 1812 is used for RADIUS authentication messages and UDP port 1813 is used for RADIUS accounting messages.

```
device(config)# radius-server host 10.20.64.208 auth-port 1812 acct-port 1813 default key secret
```

3. Create a VLAN to be used as the auth-default VLAN. This VLAN must be configured to enable authentication. When any port is enabled for 802.1X authentication or MAC authentication, the port is moved into this VLAN by default as a MAC VLAN member. Sometimes the RADIUS server may authenticate the client but not return VLAN information on where the client should be placed. The auth-default VLAN is used in this scenario.

```
device(config)# vlan 2 name auth-default-vlan
device(config-vlan-2)# exit
```

4. Create the VLANs that will be assigned to clients by RADIUS. RADIUS will return VLAN 200 for the printer. This VLAN must be active in the Brocade switch. A VLAN is active when it has at least one untagged or tagged member port. In the example, VLAN 200 is made active by adding the unused port 2/1/12 as an untagged member.

```
device(config)# vlan 200
device(config-vlan-200)# untagged ethernet 2/1/12
device(config-vlan-200)# exit
```

5. Specify which VLAN ID to use as the auth-default VLAN under authentication mode. Refer to step 3 for the use of the auth-default VLAN.

```
device(config)# authentication
device(config-authen)# auth-default-vlan 2
```

6. Enable MAC authentication on the switch under authentication mode and enable MAC authentication for port 1/1/11.

```
device(config)# authentication
device(config-authen)# mac-auth enable
device(config-authen)# mac-auth enable ethernet 1/1/11
device(config-authen)# exit
```

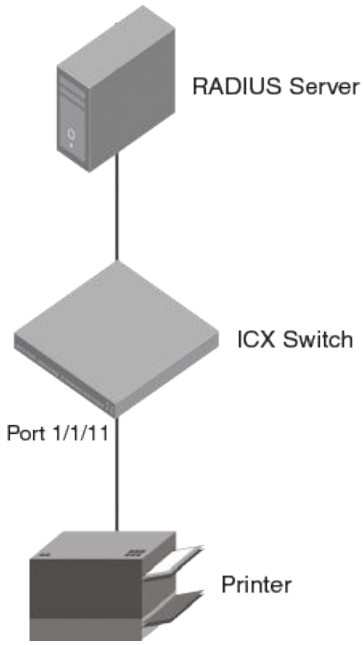
7. To verify the authentication-related configuration on the switch, use the **show running-configuration | begin authentication** command. Authentication-related configurations are stored under the keyword "authentication".

```
device# show running-configuration | begin authentication
authentication
 auth-default-vlan 2
 mac-authentication enable
 mac-authentication enable ethernet 1/1/11
!
```

Use case 3: Both 802.1X authentication and MAC authentication enabled on the same port

This use case shows the configuration required on a Brocade switch to authenticate a non-802.1X-capable client by way of MAC authentication when the client does not respond to 802.1X authentication. In the following example, the switch will attempt 802.1X authentication first. After 802.1X timeout, the switch will authenticate the printer through MAC authentication.

FIGURE 15 802.1X authentication and MAC authentication enabled on the same port



RADIUS configuration

Create a device profile for the printer's MAC address on the RADIUS server and configure the attributes in the following table.

TABLE 25 RADIUS attributes for printer profile

Attribute	Value
Tunnel-Medium-Type	802
Tunnel-Pvt-Group-ID	200
Tunnel-Type	VLAN

Brocade switch configuration

1. Specify RADIUS as an authentication server. The following command configures the switch to use the configured RADIUS server to authenticate 802.1X authentication or MAC authentication clients.

```
device(config)# aaa authentication dot1x default radius
```

2. Configure a RADIUS server. In the following example, the RADIUS server IP address is 10.20.64.208 and the shared key is "secret". The shared key should match the key given during client configuration on the RADIUS server. UDP port 1812 is used for RADIUS authentication messages and UDP port 1813 is used for RADIUS accounting messages.

```
device(config)# radius-server host 10.20.64.208 auth-port 1812 acct-port 1813 default key secret
```

3. Create a VLAN to be used as the auth-default VLAN. This VLAN must be configured to enable authentication. When any port is enabled for 802.1X authentication or MAC authentication, the port is moved into this VLAN by default as a MAC VLAN member. Sometimes the RADIUS server may authenticate the client but not return VLAN information on where the client should be placed. The auth-default VLAN is used in this scenario.

```
device(config)# vlan 2 name auth-default-vlan
device(config-vlan-2)# exit
```

4. Create the VLANs that will be assigned to clients by RADIUS. RADIUS will return VLAN 200 for the printer. This VLAN must be active in the Brocade switch. A VLAN is active when it has at least one untagged or tagged member port. In the following example, VLAN 200 is made active by adding the unused port 2/1/12 as an untagged member.

```
device(config)# vlan 200 name clientA
device(config-vlan-200)# untagged ethernet 2/1/12
device(config-vlan-200)# exit
```

5. Specify which VLAN ID to use as the auth-default VLAN under authentication mode. Refer to step 3 for the use of the auth-default VLAN.

```
device(config)# authentication
device(config-authen)# auth-default-vlan 2
```

6. Enable 802.1X on the switch under authentication mode and enable 802.1X on port 1/1/11. Configure the port control mode as **auto** in interface configuration mode. The **auto** mode enables the 802.1X authentication on the interface.

```
device(config-authen)# dot1x enable
device(config-authen)# dot1x enable ethernet 1/1/11
device(config-authen)# exit
device(config)# interface ethernet 1/1/11
device(config-if-e1000-1/1/11)# dot1x port-control auto
device(config-if-e1000-1/1/11)# exit
```

7. Enable MAC authentication on the switch under authentication mode and enable MAC authentication for port 1/1/11.

```
device(config)# authentication
device(config-authen)# mac-auth enable
device(config-authen)# mac-auth enable ethernet 1/1/11
device(config-authen)# exit
```

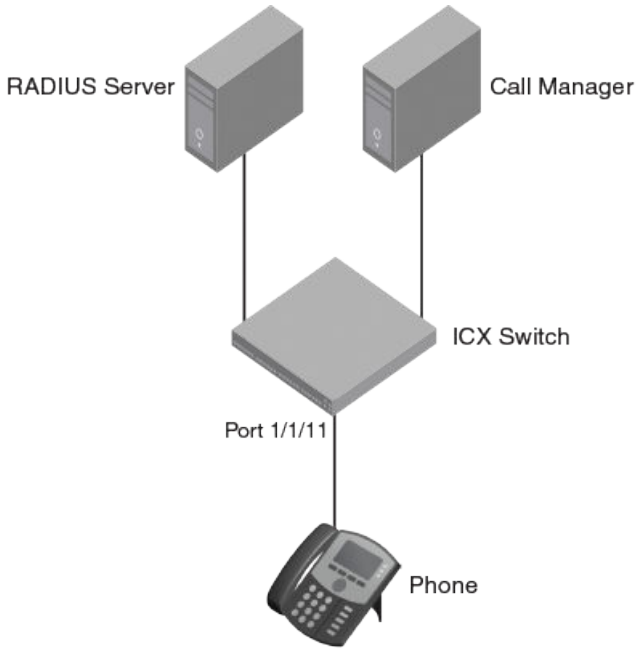
8. To verify the authentication-related configuration on the switch, use the **show running-configuration | begin authentication** command. Authentication-related configurations are stored under the keyword "authentication".

```
device# show running-configuration | begin authentication
authentication
 auth-default-vlan 2
 dot1x enable
 dot1x enable ethe 1/1/11
 mac-authentication enable
 mac-authentication enable ethe 1/1/11
!
```

Use case 4: Authenticating an IP phone using 802.1X

This use case shows the configuration required on a Brocade switch to authenticate an 802.1X-capable phone in a voice VLAN. In the following example, after authentication, the phone will be authenticated in voice VLAN 200.

FIGURE 16 Authenticating an IP phone using 802.1X



RADIUS configuration

Create a profile for the phone on the RADIUS server and configure the attributes in the following table.

TABLE 26 RADIUS attribute for IP phone

Attribute	Value	Comment
Tunnel-Medium-Type	802	
Tunnel-Pvt-Group-ID	T:200	The format is T:<Voice-VLAN-id>
Tunnel-Type	VLAN	

Brocade switch configuration

1. Specify RADIUS as an authentication server. The following command configures the switch to use the configured RADIUS server to authenticate 802.1X authentication or MAC authentication clients.

```
device(config)# aaa authentication dot1x default radius
```

2. Configure a RADIUS server. In the following example, the RADIUS server IP address is 10.20.64.208 and the shared key is "secret". The shared key should match the key given during client configuration on the RADIUS server. UDP port 1812 is used for RADIUS authentication messages and UDP port 1813 is used for RADIUS accounting messages.

```
device(config)# radius-server host 10.20.64.208 auth-port 1812 acct-port 1813 default key secret
```

3. Create a VLAN to be used as the auth-default VLAN. This VLAN must be configured to enable authentication. When any port is enabled for 802.1X authentication or MAC authentication, the port is moved into this VLAN by default as a MAC VLAN member. Sometimes the RADIUS server may authenticate the client but not return VLAN information on where the client should be placed. The auth-default VLAN is used in this scenario.

```
device(config)# vlan 2 name auth-default-vlan
device(config-vlan-2)# exit
```

4. A voice VLAN must be active in the Brocade switch. A VLAN is active when it has at least one untagged or tagged member port. The virtual interface IP of the voice VLAN will be used as a gateway IP for the phone. In the following example, VLAN 200 is used as a voice VLAN, and the unused port 1/1/1 is added as an untagged member of the VLAN.

```
device(config)# vlan 200 name voice
device(config-vlan-200)# untagged ethernet 1/1/1
device(config-vlan-200)# router-interface ve 200
device(config-vlan-200)# exit
device(config)# interface ve 200
device(config-vif-200)# ip address 172.20.74.1/24
```

5. Configure the IP helper on the virtual interface of the voice VLAN so that a DHCP request from the IP phone is forwarded to the call manager. In the following topology, the call manager IP address is 10.20.74.31.

```
device(config-vlan-200)# ip helper-address 1 10.20.74.31
device(config-vif-200)# exit
```

6. LLDP will be configured automatically if the Brocade vendor-specific attribute "Foundry-Voice-Phone-Config" is configured in the RADIUS server. LLDP advertises the VLAN information so that the client connected to the port learns the voice VLAN.

   If the Brocade vendor-specific attribute "Foundry-Voice-Phone-Config" is not configured, configure LLDP manually to advertise VLAN 200 as the voice VLAN on port 1/1/11. An LLDP warning message will be shown to indicate that port 1/1/11 is not part of VLAN 200. This warning can be ignored, as port 1/1/11 will be added to VLAN 200 by way of dynamic VLAN assignment after authentication.

```
device(config)# lldp run
device(config)# lldp med network-policy application voice tagged vlan 200 priority 5 dscp 46 ports
ethernet 1/1/11
LLDP Warning: Network policy: Port 1/1/11 is not a member of VLAN 200
```

7. Specify which VLAN ID to use as the auth-default VLAN under authentication mode. Refer to step 3 for the use of the auth-default VLAN.

```
device(config)# authentication
device(config-authen)# auth-default-vlan 2
```

8. Enable 802.1X on the switch under authentication mode and enable 802.1X on port 1/1/11. Configure the port control mode as **auto** in interface configuration mode. The **auto** mode enables the 802.1X authentication on the interface.

```
device(config)# authentication
device(config-authen)# dot1x enable
device(config-authen)# dot1x enable ethernet 1/1/11
device(config-authen)# exit
device(config)# interface ethernet 1/1/11
device(config-if-e1000-1/1/11)# dot1x port-control auto
device(config-if-e1000-1/1/11)# exit
```

9. Enable PoE on port 1/1/11 using the **inline power** command in interface configuration mode.

```
device(config)# interface ethernet 1/1/11
device(config-if-e1000-1/1/11)# inline power
device(config-if-e1000-1/1/11)# exit
```

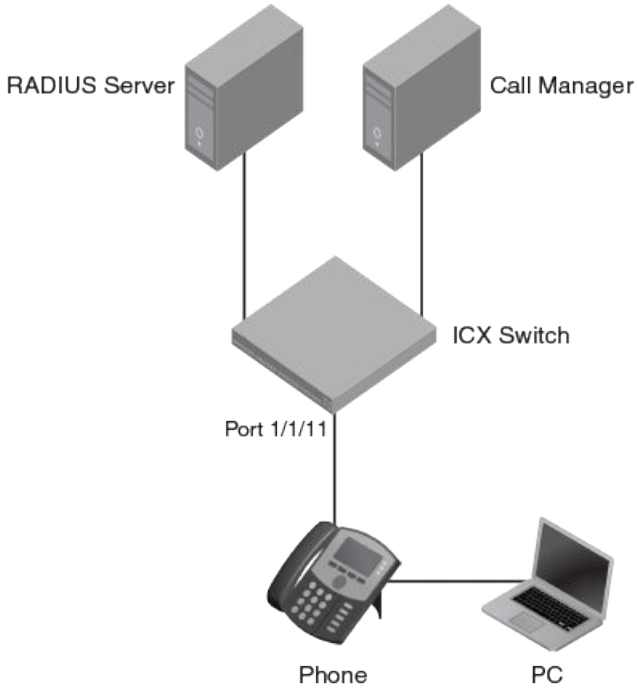
10. To verify the authentication-related configuration on the switch, use the **show running-configuration | begin authentication** command. Authentication-related configurations are stored under the keyword "authentication".

```
device# show running-configuration | begin authentication
authentication
 auth-default-vlan 2
 dot1x enable
 dot1x enable ethe 1/1/11
!
```

Use case 5: Authenticating an 802.1X phone and an 802.1X PC on the same port

This use case shows the configuration required on a Brocade switch to authenticate an 802.1X-capable phone in the voice VLAN and an 802.1X-capable PC in the data VLAN on the same port. In the following example, after authentication, the phone will be authenticated in voice VLAN 200 and the PC will be authenticated in data VLAN 201.

FIGURE 17 Authenticating an 802.1X phone and an 802.1X PC on the same port



RADIUS configuration

Create a profile for the phone on the RADIUS server and configure the attributes in the following table.

TABLE 27 RADIUS attribute for phone

Attribute	Value	Comment
Tunnel-Medium-Type	802	
Tunnel-Pvt-Group-ID	T:200	The format is T:<Voice-VLAN-id>
Tunnel-Type	VLAN	

Create a user profile on the RADIUS server and configure the attributes in the following table.

TABLE 28 RADIUS attributes for PC user

Attribute	Value
Tunnel-Medium-Type	802
Tunnel-Pvt-Group-ID	201
Tunnel-Type	VLAN

Brocade switch configuration

1. Specify RADIUS as an authentication server. The following command configures the switch to use the configured RADIUS server to authenticate 802.1X authentication or MAC authentication clients.

```
device(config)# aaa authentication dot1x default radius
```

2. Configure a RADIUS server. In the following example, the RADIUS server IP address is 10.20.64.208 and the shared key is "secret". The shared key should match the key given during client configuration on the RADIUS server. UDP port 1812 is used for RADIUS authentication messages and UDP port 1813 is used for RADIUS accounting messages.

```
device(config)# radius-server host 10.20.64.208 auth-port 1812 acct-port 1813 default key secret
```

3. Create a VLAN to be used as the auth-default VLAN. This VLAN must be configured to enable authentication. When any port is enabled for 802.1X authentication or MAC authentication, the port is moved into this VLAN by default as a MAC VLAN member. Sometimes the RADIUS server may authenticate the client but not return VLAN information on where the client should be placed. The auth-default VLAN is used in this scenario.

```
device(config)# vlan 2 name auth-default-vlan
device(config-vlan-2)# exit
```

4. Create the VLANs that will be assigned to clients by RADIUS. RADIUS will return VLAN 201 for the PC and this VLAN must be active in the Brocade switch. A VLAN is active when it has at least one untagged or tagged member port. In te following example, VLAN 201 is made active by adding the unused port 2/1/12 as an untagged member.

```
device(config)# vlan 201 name data
device(config-vlan-201)# untagged ethernet 2/1/12
device(config-vlan-201)# exit
```

5. The voice VLAN must be active in the Brocade switch. A VLAN is active when it has at least one untagged or tagged member port. The virtual interface IP of the voice VLAN will be used as the gateway IP for the phone. In the following example, VLAN 200 is used as the voice VLAN, and the unused port 1/1/1 is added as an untagged member of the VLAN.

```
device(config)# vlan 200 name voice
device(config-vlan-200)#untagged ethernet 1/1/1
device(config-vlan-200)#router-interface ve 200
device(config-vlan-200)#exit
device(config)#interface ve 200
device(config-vif-200)#ip address 172.20.74.1/24
```

6. Configure the IP helper on the virtual interface of the voice VLAN so that the DHCP request from the IP phone is forwarded to the call manager. In the following topology, the call manager IP address is 10.20.74.31.

```
device(config-vif-200)# ip helper-address 1 10.20.74.31
device(config-vif-200)# exit
```

7. LLDP will be automatically configured if the Brocade vendor-specific attribute "Foundry-Voice-Phone-Config" is configured in the RADIUS server. LLDP advertises the VLAN information so that the client connected to the port learns the voice VLAN.

   If the Brocade vendor-specific attribute "Foundry-Voice-Phone-Config" is not configured, configure LLDP manually to advertise VLAN 200 as the voice VLAN on port 1/1/11. An LLDP warning message will be shown to indicate that port 1/1/11 is not part of VLAN 200. This warning can be ignored, as port 1/1/11 will be added to VLAN 200 by way of dynamic VLAN assignment after authentication.

   ```
   device(config)# lldp run
   device(config)# lldp med network-policy application voice tagged vlan 200 priority 5 dscp 46 ports
   ethernet 1/1/11
   LLDP Warning: Network policy: Port 1/1/11 is not a member of VLAN 200
   ```

8. Specify which VLAN ID to use as the auth-default VLAN under authentication mode. Refer to step 3 for the use of as the auth-default VLAN.

   ```
   device(config)# authentication
   device(config-authen)# auth-default-vlan 2
   ```

9. Enable 802.1X on the switch under authentication mode and enable 802.1X on port 1/1/11. Configure the port control mode as **auto** in interface configuration mode. The **auto** mode enables the 802.1X authentication on the interface.

   ```
   device(config)# authentication
   device(config-authen)# dot1x enable
   device(config-authen)# dot1x enable ethernet 1/1/11
   device(config-authen)# exit
   device(config)# interface ethernet 1/1/11
   device(config-if-e1000-1/1/11)# dot1x port-control auto
   device(config-if-e1000-1/1/11)# exit
   ```

10. Enable PoE on port 1/1/11 using the **inline power** command in interface configuration mode.

    ```
    device(config)# interface ethernet 1/1/11
    device(config-if-e1000-1/1/11)# inline power
    device(config-if-e1000-1/1/11)# exit
    ```

11. To verify the authentication-related configuration on the switch, use the **show running-configuration | begin authentication** command. Authentication-related configurations are stored under the keyword "authentication".

    ```
    device# show running-configuration | begin authentication
    authentication
     auth-default-vlan 2
     dot1x enable
     dot1x enable ethe 1/1/11
    !
    ```

Displaying 802.1X information

You can display the following 802.1X-related information:

- The 802.1X configuration on the device and on individual ports
- Statistics about the EAPOL frames passing through the device
- 802.1X-enabled ports dynamically assigned to a VLAN
- User-defined and dynamically applied MAC address filters and IP ACLs currently active on the device
- The 802.1X multiple-host configuration

Displaying 802.1X statistics

To display 802.1X statistics for an individual port, enter the **show dot1x statistics** command.

```
device# show dot1x statistics ethernet 1/1/1
Port 1/1/1 Statistics:
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
```

The following table describes the information displayed by the **show dot1x statistics** command for an interface.

TABLE 29 Output from the show dot1x 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.

Displaying dynamically-assigned VLAN information

The output of the **show vlan ethernet** command displays dynamically assigned VLAN information.

```
device# show vlan ethernet 2/1/1

  Total PORT-VLAN entries: 14
Maximum PORT-VLAN entries: 4095

Legend: [Stk=Stack-Id, S=Slot]

PORT-VLAN 2009, Name [None], Priority level0, Spanning tree Off
Untagged Ports: None
   Tagged Ports: (U1/M1)  48
   Tagged Ports: (U2/M1)   2
   Uplink Ports: None
DualMode Ports: None
```

```
Mac-Vlan Ports: (U2/M1)   1
     Monitoring: Disabled
```

Displaying information about MAC address filters and IP ACLs

You can display information about currently active user-defined and dynamically applied MAC address filters and IP ACLs.

Displaying user-defined MAC address filters and IP ACLs

To display the user-defined MAC address filters active on the device, enter the following command.

```
device# show dot1x mac-filter
Port 1/3 (User defined MAC Address Filter) :
mac filter 1 permit any any
```

To display the user-defined IP ACLs active on the device, enter the following command.

```
device# show dot1x ip-ACL
Port 1/3 (User defined IP ACLs):
Extended IP access list Port_1/3_E_IN
permit udp any any
```

Displaying dynamically applied MAC address filters and IP ACLs

To display the dynamically applied MAC address filters active on an interface, enter a command such as the following.

```
device# show dot1x mac-filter ethernet 2/1/11

802.1X MAC Address Filter Information :

Port 2/1/11:
Dynamic MAC filter-list: 1
```

To display the dynamically applied IP ACLs active on an interface, enter a command such as the following.

```
device# show dot1x ip-acl ethernet 2/1/11
802.1X IP ACL Information :

Port 2/1/11 : 0022.0002.0002
In-bound IP ACL : 100
```

Displaying configuration of 802.1X ports

The output of the **show dot1x configuration** command indicates the configuration details of 802.1X ports.

To display the configuration details of 802.1X ports globally on the device, enter the **show dot1x configuration** command.

```
device# show dot1x configuration
PAE Capability                 : Authenticator Only
Status                         : Enabled
Auth Order                     : mac-auth dot1x
Default VLAN                   : 2
Restricted VLAN                : 4
Critical VLAN                  : 3
Guest VLAN                     : 5
Action on Auth failure         : Move to Restricted VLAN (4)
MAC Session Aging              : Enabled
Filter Strict Security         : Enabled
Re-authentication              : Disabled
Session max sw-age             : 120 seconds
Session max hw-age             : 70 seconds
Quiet-period                   : 60 seconds
TX-period                      : 30 seconds
```

```
Reauth-period                  : 60 seconds
Supplicant-timeout             : 30 seconds
Max Reauth requests            : 2
Protocol Version               : 1
Mixed-STK#
```

To display the configuration details of 802.1X ports on an interface, enter a command such as the following.

```
device# show dot1x configuration ethernet 1/1/1
Port 1/1/1 Configuration:
Port-Control                   : control-auto
Auth Order                     : mac-auth dot1x
Action on Auth failure         : Move to Restricted VLAN (4)
Action on Auth timeout         : Treat as a failed authentication
Action on Voice timeout        : Treat as a failed authentication
Filter Strict Security         : Enabled
DoS Protection                 : Disabled (limit = 512)
Source-guard Protection        : Disabled
Reauth-timeout                 : 60 seconds
Aging                          : Enabled
Max-sessions                   : 4
```

Displaying the 802.1X authentication sessions

Use the **show dot1x sessions** command to view details of the 802.1X authentication sessions, such as the ports, MAC addresses, IP addresses, VLANs, and so on.

NOTE
The IP address of the authenticated host is displayed only if an IP ACL is applied to the interface based on the RADIUS server response.

NOTE
The IP source guard ACL entry is displayed only if the IP Source Guard Protection is enabled on the port.

The following example displays 802.1X sessions for all interfaces.

```
device(config)# show dot1x sessions all
-------------------------------------------------------------------------------
Port   MAC             IP          User    Vlan  Auth     ACL     Age   PAE
       Addr            Addr        Name          State                  State
-------------------------------------------------------------------------------
2/1/1  0010.9400.1303  192.85.1.2   User1  200   permit  IPSG    Ena   AUTHENTICATED
2/1/1  0010.9400.1304  1.1.1.4      User2  2009  permit  IPSG    Ena   AUTHENTICATED
2/1/1  0010.9400.1305  1.1.1.2      User3  2009  permit  IPSG    Ena   AUTHENTICATED
2/1/1  0010.9400.1306  1.1.1.6      User4  2009  permit  IPSG    Ena   AUTHENTICATED
```

The following example displays 802.1X sessions for a specified interface.

```
device(config)# show dot1x sessions ethernet 2/1/1
-------------------------------------------------------------------------------
Port   MAC             IP          User    Vlan  Auth     ACL     Age   PAE
       Addr            Addr        Name          State                  State
-------------------------------------------------------------------------------
2/1/1  0010.9400.1303  192.85.1.2   User1  200   permit  IPSG    Ena   AUTHENTICATED
```

The following example displays a brief description of 802.1X sessions.

```
device# show dot1x sessions brief
-------------------------------------------------------------------------------
Port    Number of   Number of   Number of   Untagged          Dynamic   Dynamic
        Attempted   Authorized  Denied      VLAN Type         Port ACL  MAC-Filt
        Users       Users       Users
-------------------------------------------------------------------------------
1/1/2   1           1           0           Radius-VLAN        No       No
1/1/3   0           0           0           Auth-Default-VLAN  No       No
1/1/4   0           0           0           Auth-Default-VLAN  No       No
1/1/5   0           0           0           Auth-Default-VLAN  No       No
2/1/1   0           0           0           Auth-Default-VLAN  No       No
2/1/2   0           0           0           Auth-Default-VLAN  No       No
2/1/4   0           0           0           Auth-Default-VLAN  No       No
```

Displaying MAC authentication information

You can display the following information about the MAC authentication configuration:

- Information about authenticated MAC addresses
- Information about the MAC authentication configuration
- Authentication Information for a specific MAC address or port
- MAC authentication settings and authenticated MAC addresses for each port where MAC authentication is enabled
- The MAC addresses that have been successfully authenticated
- The MAC addresses for which authentication was not successful

Displaying the MAC authentication sessions

Use the **show mac-authentication sessions** command to view details of the MAC authentication sessions, such as the ports, MAC addresses, IP addresses, VLANs, and so on.

NOTE
The IP address of the authenticated host is displayed only if an IP ACL is applied to the interface based on the RADIUS server response.

NOTE
The IP source guard ACL entry is displayed only if the IP Source Guard Protection is enabled on the port.

The following example displays MAC authentication sessions for all interfaces.

```
device# show mac-authentication sessions all
-------------------------------------------------------------------------------
Port        MAC             IP              Vlan  Auth      ACL     Age
            Addr            Addr                  State
-------------------------------------------------------------------------------
1/1/2       0010.94ab.0021  192.85.1.2      300   Yes       IPSG    Ena
1/1/3       0110.94ab.0021  1.1.1.4         3     Yes       IPSG    Ena
```

The following example displays MAC authentication sessions for a specified interface.

```
device# show mac-authentication sessions ethernet 1/1/2
-------------------------------------------------------------------------------
Port        MAC             IP              Vlan  Auth      ACL     Age
            Addr            Addr                  State
-------------------------------------------------------------------------------
1/1/2       0010.94ab.0021  192.85.1.2      300   Yes       IPSG    Ena
```

The following example displays a brief description of MAC authentication sessions.

```
device# show mac-authentication sessions brief
-----------------------------------------------------------------------------------------------
Port      Number of           Number of          Number of         Untagged          Dynamic
          Attempted Users     Authorized Users   Denied Users      VLAN Type         Port ACL
-----------------------------------------------------------------------------------------------
1/1/2     1                   1                  0                 Radius-VLAN         No
1/1/3     0                   0                  0                 Auth-Default-VLAN   No
1/1/4     0                   0                  0                 Auth-Default-VLAN   No
1/1/5     0                   0                  0                 Auth-Default-VLAN   No
2/1/1     0                   0                  0                 Auth-Default-VLAN   No
2/1/2     0                   0                  0                 Auth-Default-VLAN   No
2/1/4     0                   0                  0                 Auth-Default-VLAN   No
```

Clearing 802.1X details

You can clear the 802.1X sessions and statistics counters on individual interfaces or on a range of interfaces.

To clear the 802.1X statistics counters on an interface, enter a command such as the following.

```
device# clear dot1x statistics ethernet 1/1/1
```

To clear the 802.1X statistics counters on a range of interfaces, enter a command such as the following.

```
device# clear dot1x statistics ethernet 1/1/1 to 1/1/10
```

To clear the 802.1X sessions for a specific MAC address, enter a command such as the following.

```
device# clear dot1x sessions  0000.0034.abd4
```

To clear the 802.1X sessions on an interface, enter a command such as the following.

```
device# clear dot1x sessions ethernet 1/1/1
```

To clear the 802.1X sessions on a range of interfaces, enter a command such as the following.

```
device# clear dot1x sessions ethernet 1/1/1 to 1/1/8
```

Clearing MAC authentication details

You can clear the MAC authentication sessions and statistics counters on individual interfaces or on a range of interfaces.

To clear the MAC authentication statistics counters on an interface, enter the command such as the following.

```
device# clear mac-authentication statistics ethernet 1/1/1
```

To clear the MAC authentication statistics counters on a range of interfaces, enter a command such as the following.

```
device# clear mac-authentication statistics ethernet 1/1/1 to 1/1/10
```

To clear the MAC authentication sessions for a specific MAC address, enter a command such as the following.

```
device# clear mac-authentication sessions 0000.0034.abd4
```

To clear the MAC authentication sessions on an interface, enter a command such as the following.

```
device# clear mac-authentication sessions ethernet 1/1/1
```

To clear the MAC authentication sessions on a range of interfaces, enter a command such as the following.

```
device# clear mac-authentication sessions ethernet 1/1/1 to 1/1/8
```

HTTP and HTTPS

Web Authentication using HTTP or HTTPS services

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 using HTTP or HTTPS services method provides an ideal port-based authentication alternative to MAC authentication without the complexities and cost of 802.1x authentication. Hosts 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 host browser is directed to an authentication web page on the FastIron switch. The web page prompts the host to enter a username and password or a passcode. The credentials a host enters are used by a trusted source to authenticate the host MAC address. (Multiple MAC addresses can be authenticated with the same username and password.)

If the authentication is unsuccessful, you are 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 user is directed to the requested URL.

While a MAC address is in the authenticated state, the host can forward data through the FastIron 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 the “Specifying hosts that are permanently authenticated” section.)
- The re-authentication timer expires and the host is required to re-authenticate (Refer to the “Configuring the re-authentication period” section).

- The host has remained inactive for a period of time and the inactive period timer has expired. (Refer to the “Forcing re-authentication after an inactive period” section.)
- 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 the “Forcing re-authentication when ports are down” section.)
- The authenticated client is cleared from the Web Authentication table. (Refer to the Clearing authenticated hosts from the web authentication table” section.)

The FastIron switch can be configured to automatically authenticate a host MAC address. The host will not be required to log in 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 Web Authentication 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.

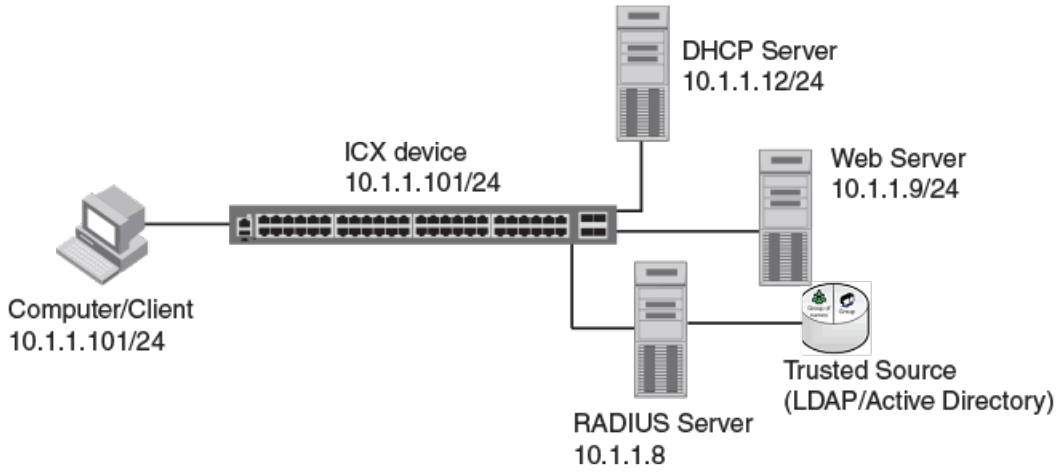
The basic topology of a network in which Web Authentication is used requires the following components:

- A Ruckus FastIron switch running a software release that supports Web Authentication
- A DHCP server, if dynamic IP addressing is to be used
- A computer or host with a web browser

Your configuration may also require a RADIUS server with a trusted source such as LDAP or Active Directory.

NOTE
The web server, RADIUS server, and DHCP server can all be the same server.

FIGURE 18 Basic network topology for Web Authentication



Captive Portal user authentication (external Web Authentication)

Captive Portal user authentication provides a means to authenticate the 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 the Aruba ClearPass server (external server) that is integrated with the RADIUS server.

> **NOTE**
> Because the authentication server and web login page reside in an external server, Captive Portal user authentication is referred to as external Web Authentication in this document.

To equip the Brocade switch to handle the HTTP redirection mechanism, configuration details specific to the Aruba ClearPass server such as virtual IP address, HTTP or HTTPS protocol port number, and login page details hosted on the Aruba ClearPass server must be specified on the switch. Upon receiving the redirected web access request, Aruba ClearPass server honors the login page to the client which in turn submits the user login credentials. The Aruba ClearPass 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 more details for configuring external captive portal on 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.

The Brocade switch makes use of the credentials for initiating the authentication process through the RADIUS server, which is integrated with Aruba ClearPass server.

> **NOTE**
> The RADIUS server on the Brocade switch and the one integrated with the Aruba ClearPass 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 277 and Defining the Web Authentication cycle on page 277.

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 Aruba ClearPass server.

The details configured in the Aruba Captive Portal profile enable the switch to handle the HTTP redirection mechanism and redirect the client to the login page hosted on the Aruba ClearPass server. The Captive Portal profile is then applied on an external Web Authentication-enabled VLAN.

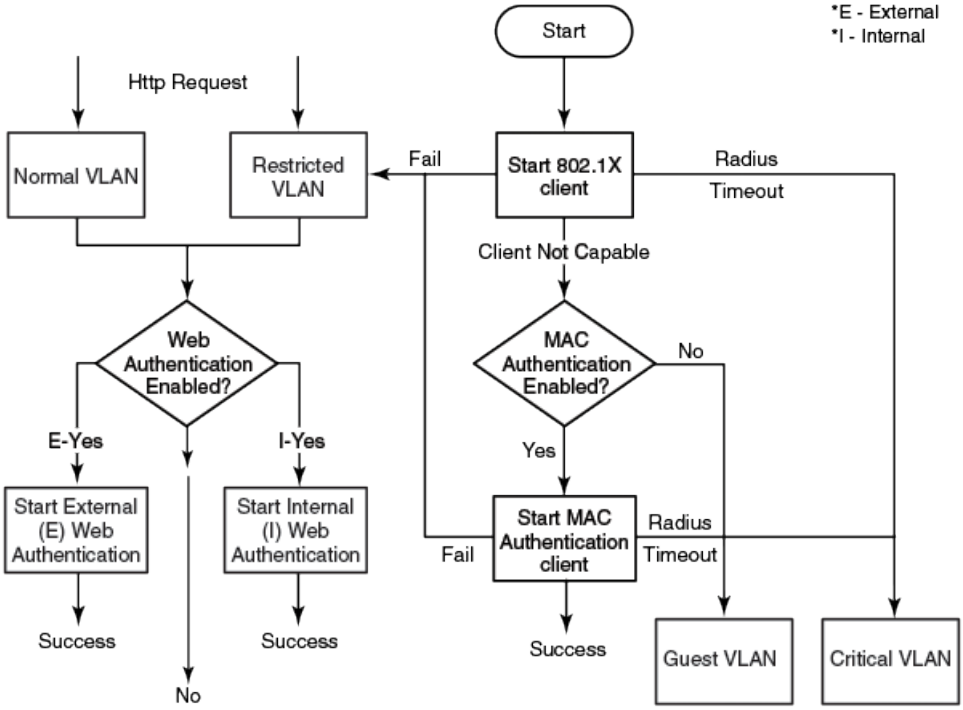
External Web Authentication on a VLAN

External Web Authentication can be configured as a fallback authentication method for Flexible authentication (a combination of 802.1X authentication and MAC authentication).

External Web Authentication can be enabled on any VLAN associated with Flexible authentication (restricted VLAN, guest VLAN, or critical VLAN). You can also enable external Web Authentication 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 external Web Authentication 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 External Web Authentication 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.

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 and MAC address filters 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 30 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 Brocade switch.

The following table shows the syntax for specifying the IP ACLs on the RADIUS server.

TABLE 31 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 32 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 services are enabled on the device. Web management must be enabled for HTTP and HTTPS access.
- Web Authentication works only on the default HTTP or HTTPS port.
- If the secure Web server is used, generate a crypto SSL certificate or import digital certificates issued by a third-party Certificate Authority (CA) to access a secure Web page.
- 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 an MCT VLAN.
- External and local or internal Web Authentication cannot be configured on the same VLAN.
- External and local or internal Web Authentication uses RADIUS as the authentication method.

The following requirement 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 requirements apply to Web Authentication in Layer 3 images:

- Each Web Authentication VLAN must have a virtual interface (VE).
- The VE must have at least one assigned IPv4 address.

When 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.

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
   device(config)# radius-server key $GSig@U\
   ```

   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 268.

7. Configure other Web Authentication options (refer to Web Authentication options configuration on page 276).

Prerequisites for external Web Authentication

The following are the prerequisites to support external Web Authentication on Brocade ICX switches.

- 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 33 Mandatory parameters to be added on the Aruba ClearPass server

Fields	Value	Description
Submit URL	http://<IP address>/Forms/webauth_cpss	IP address is the IP address of the NAS device. Specifies the URL of the NAS device's login form.
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.
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.brocade.com or http://www.brocade.com	Specifies the destination URL to which the client is redirected after authentication.

Other vendor-specific details are selected by default.

Prerequisite configurations on ICX device for external web authentication

The following are the prerequisites to support external Web Authentication on 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 Aruba ClearPass server such as virtual IP address, http or https protocol port number, and login-page details hosted on the Aruba ClearPass 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
   device(config)#
   ```

2. Enter the **captive-portal** command to create a user-defined Captive Portal profile.

   ```
   device(config)# captive-portal cp_brocade
   device (config-cp-cp_brocade)#
   ```

   The Captive Portal command mode is enabled, where you can specify the Aruba ClearPass server details that enable the switch to handle HTTP redirection mechanism.

3. Enter the **virtual-ip** command to configure the IP address of the Aruba ClearPass server as the virtual IP address.

   ```
   device (config-cp-cp_brocade)#  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.

   ```
   device (config-cp-cp_brocade)#  virtual-port 80
   ```

   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.

   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 Aruba ClearPass server.

   ```
   device (config-cp-cp_brocade)#  login-page brocadeguestlogin.php
   ```

   The login page details must be same as the login page hosted on the Aruba ClearPass server.

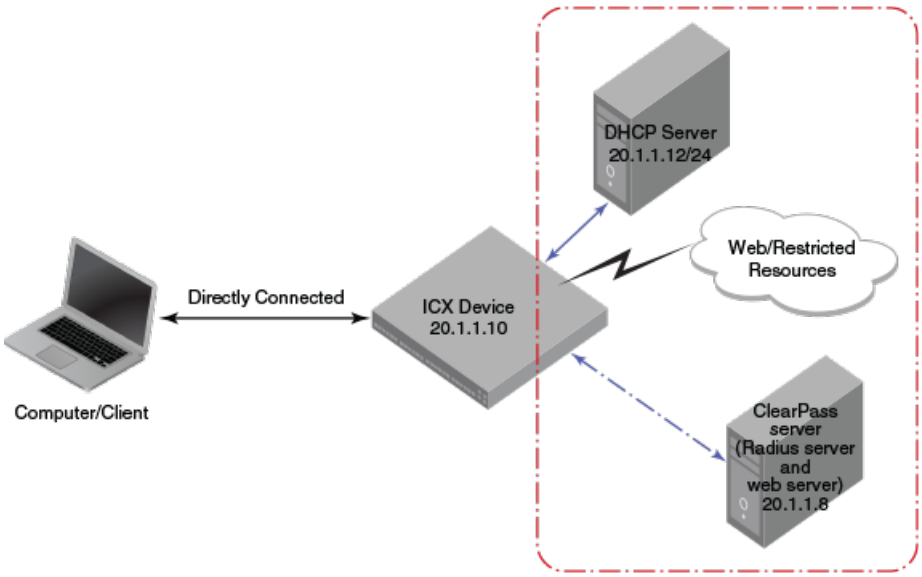
6. (Optional) Enter the **show captive-portal** command to view the output of the configured Captive Portal profile.

```
device(config)# show captive-portal cp-brocade
Configured Captive Portal Profile Details :
  cp-name                :cp-brocade
  virtual-ip             :10.21.240.42
  virtual-port           :80
  user-role              :guest
  login-page             :brocadeguestlogin.php
```

Configuring external Web Authentication

The Captive Portal profile must be created to attach it to the Web Authentication-enabled VLAN. For more information, refer to Creating the Captive Portal profile for external Web Authentication on page 265.

FIGURE 20 Basic network topology for 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, Aruba ClearPass server, and other servers.
   - On a Layer 2 switch, make sure the FastIron 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 Aruba ClearPass 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
   device(config)# radius-server key $GSig@U\
   ```

   NOTE
   The RADIUS key configured should be the same as the key configured in the Aruba ClearPass server.

   NOTE
   A RADIUS server other than the one integrated with Aruba ClearPass server can also be configured.

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_brocade
   ```

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 Aruba ClearPass 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 switch to use one of four Web Authentication modes:

- Username and password: Blocks users from accessing the switch until they enter a valid username and password on a web login page. Refer to Using local user databases on page 268.
- Passcode: Blocks users from accessing the switch until they enter a valid passcode on a web login page. Refer to Passcodes for user authentication on page 271.
- 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 Automatic authentication on page 275.

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 269.

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 291.

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 270.

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. To configure the FastIron switch to use a RADIUS server, refer to the RADIUS security on page 59. 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 271 or Setting the Web Authentication failover sequence on page 271

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 292.

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 274).

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 276). 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 280 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 3
device(config-vlan-10-webauth)# trust port ethernet 6 to 10
```

The command examples configure ports 3 and ports 6 to 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.

```
Brocade(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.

```
Brocade(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 *Brocade 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

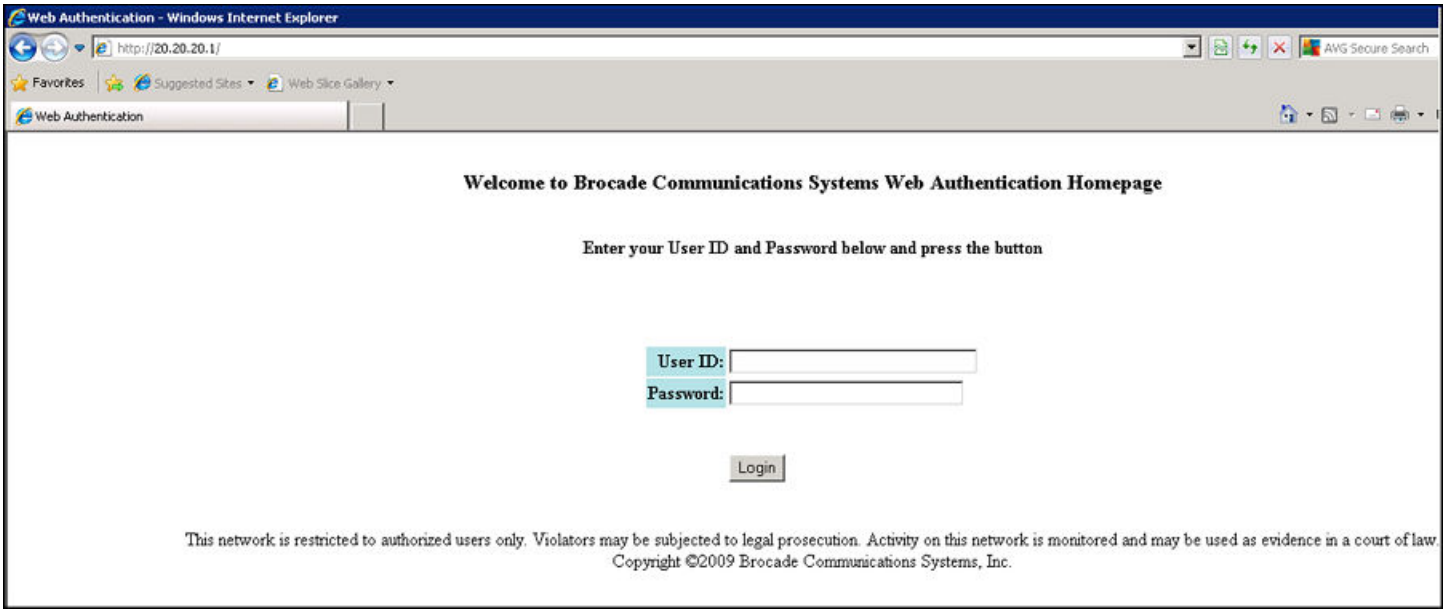
Several different web pages may be displayed during Web Authentication.

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 280).

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.

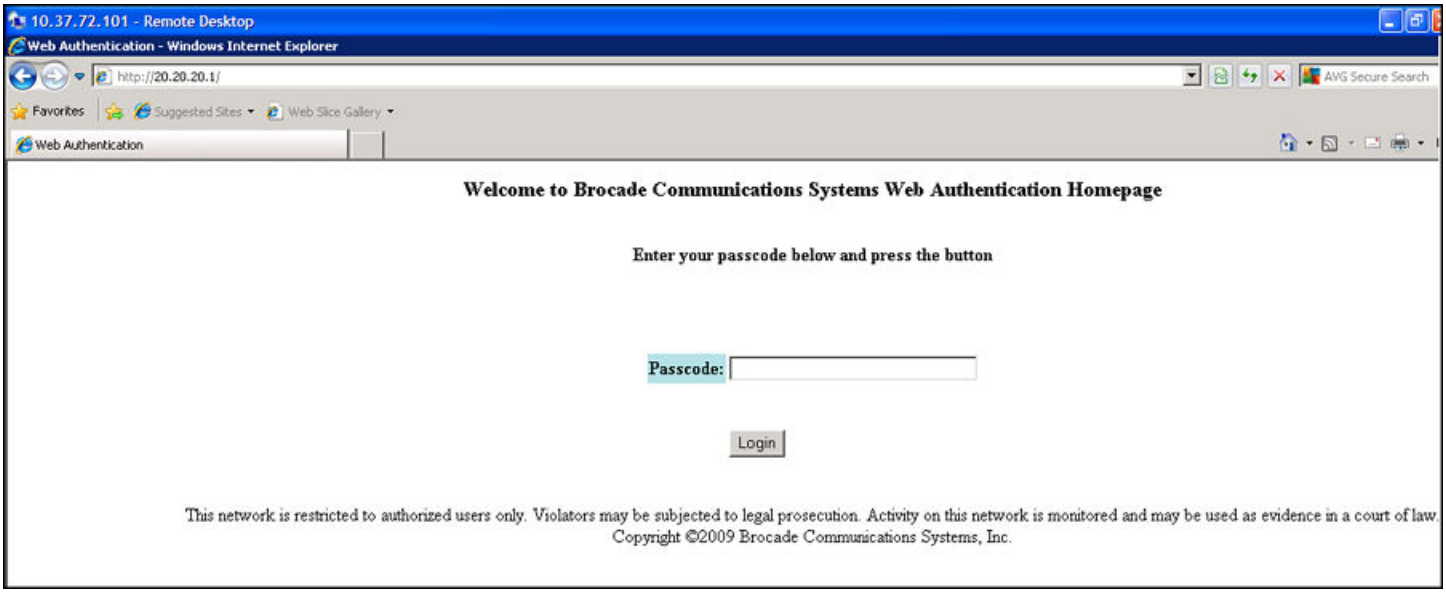
FIGURE 21 Example of login page when automatic authentication is disabled and a local user database is enabled



The user enters a username and password, which are sent for authentication.

If passcode authentication is enabled, the following login page appears.

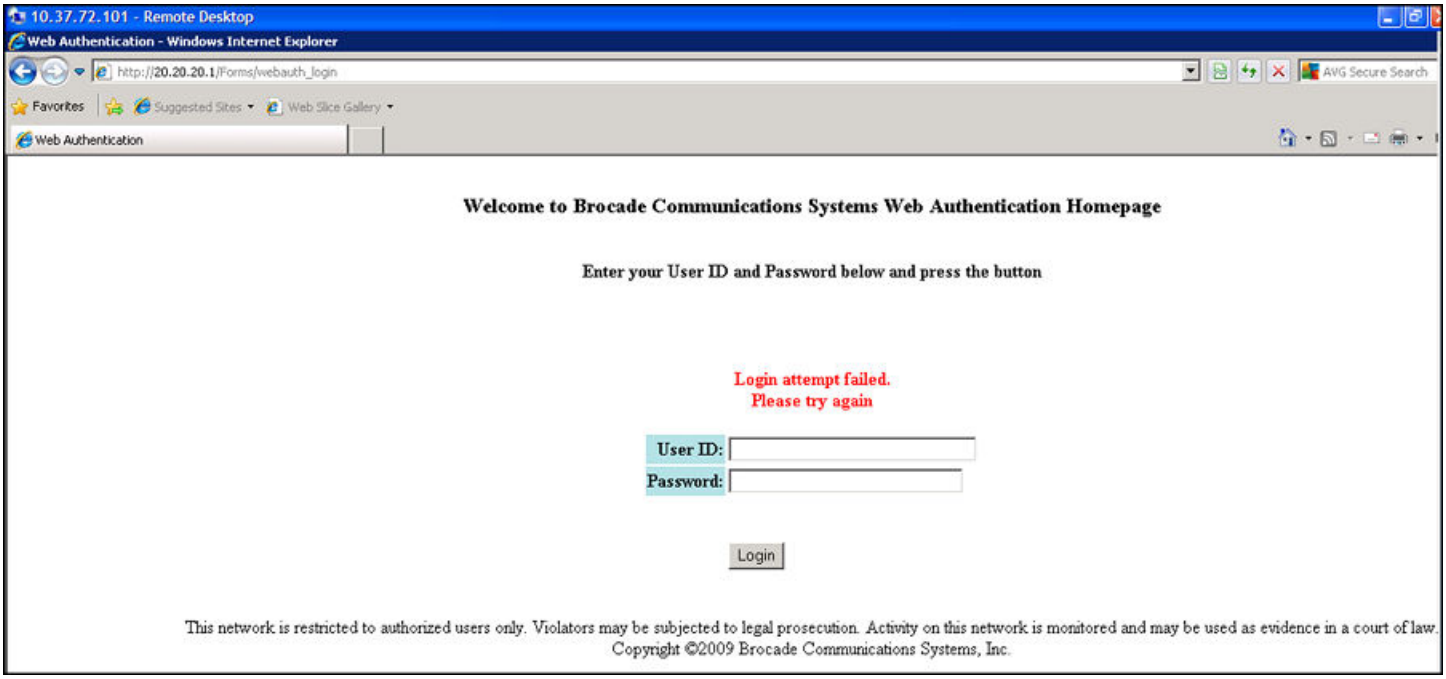
FIGURE 22 Example of login page when automatic authentication is disabled and passcode authentication is enabled



The user enters a passcode, which is sent for authentication.

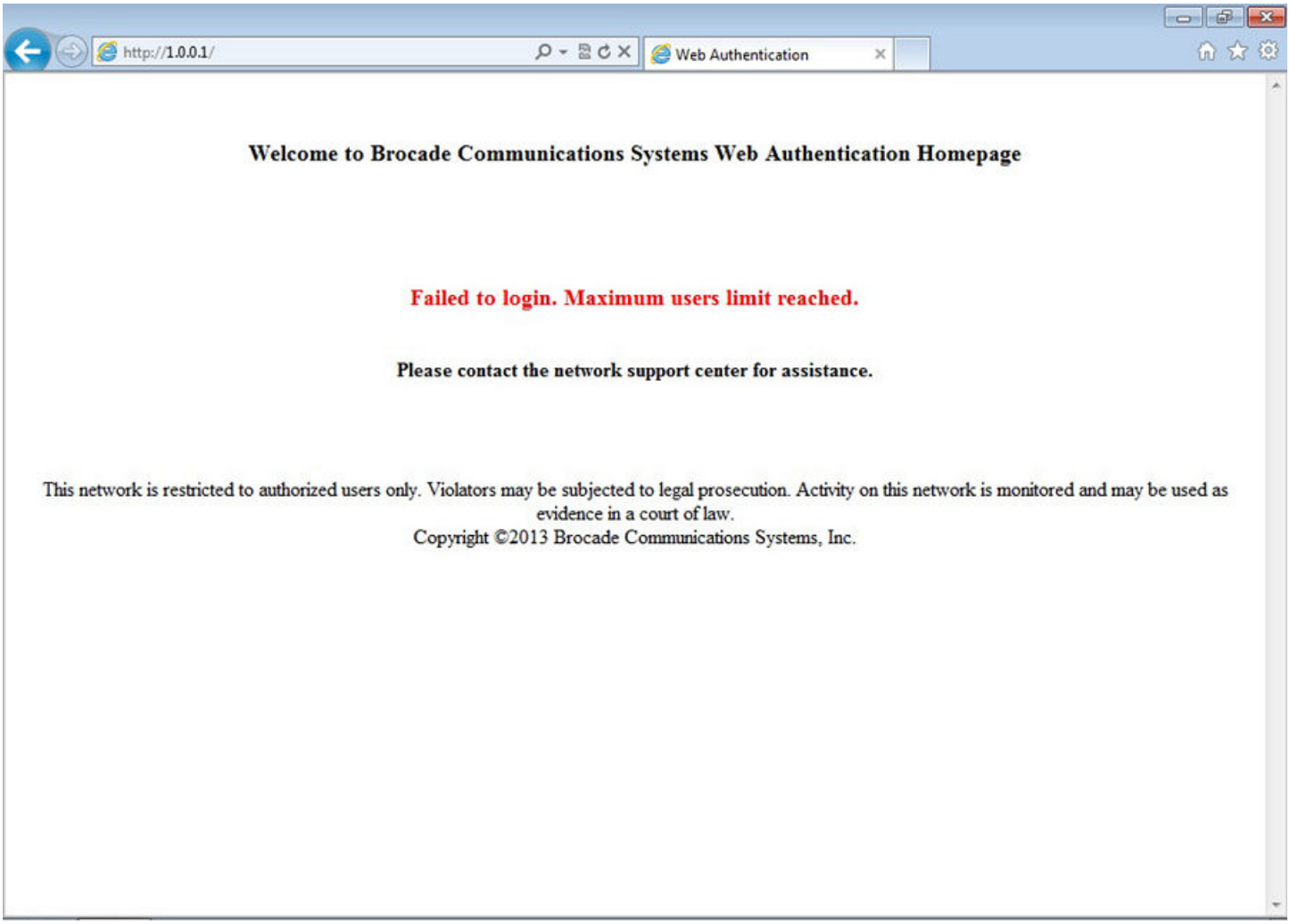
If Web Authentication fails, the following try again page appears.

FIGURE 23 Example of a try again page



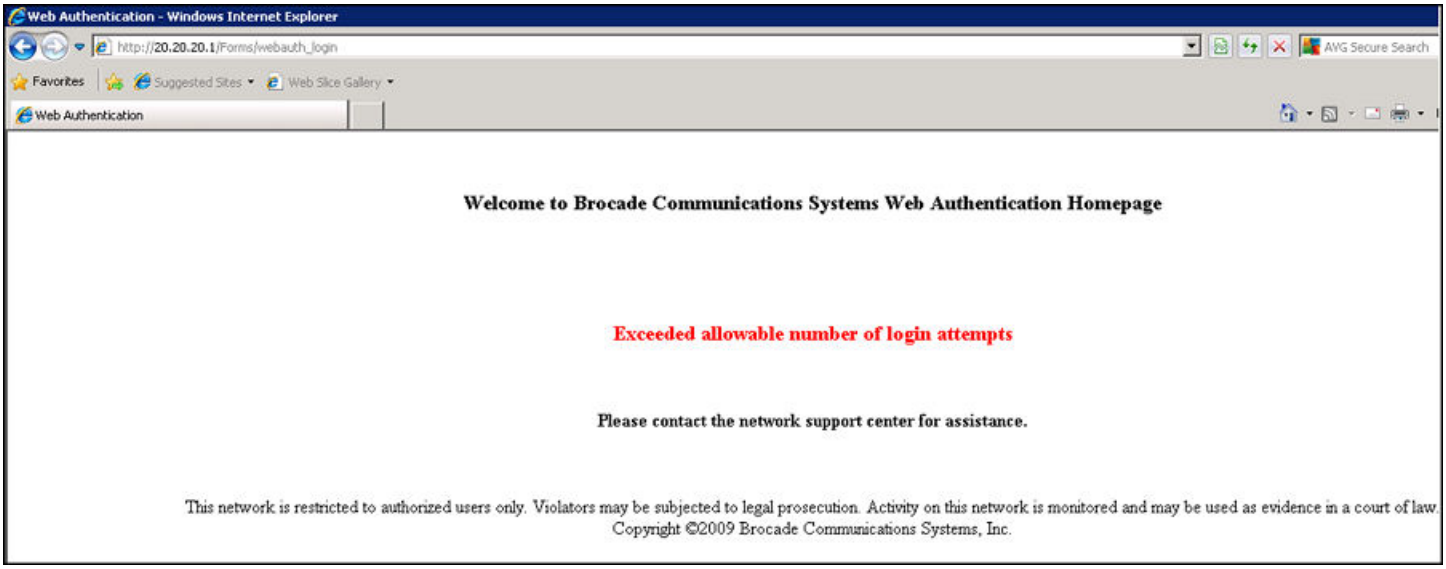
If the limit for the number of authenticated users on the network is exceeded, the following maximum host limit page appears.

FIGURE 24 Example of a maximum host limit page



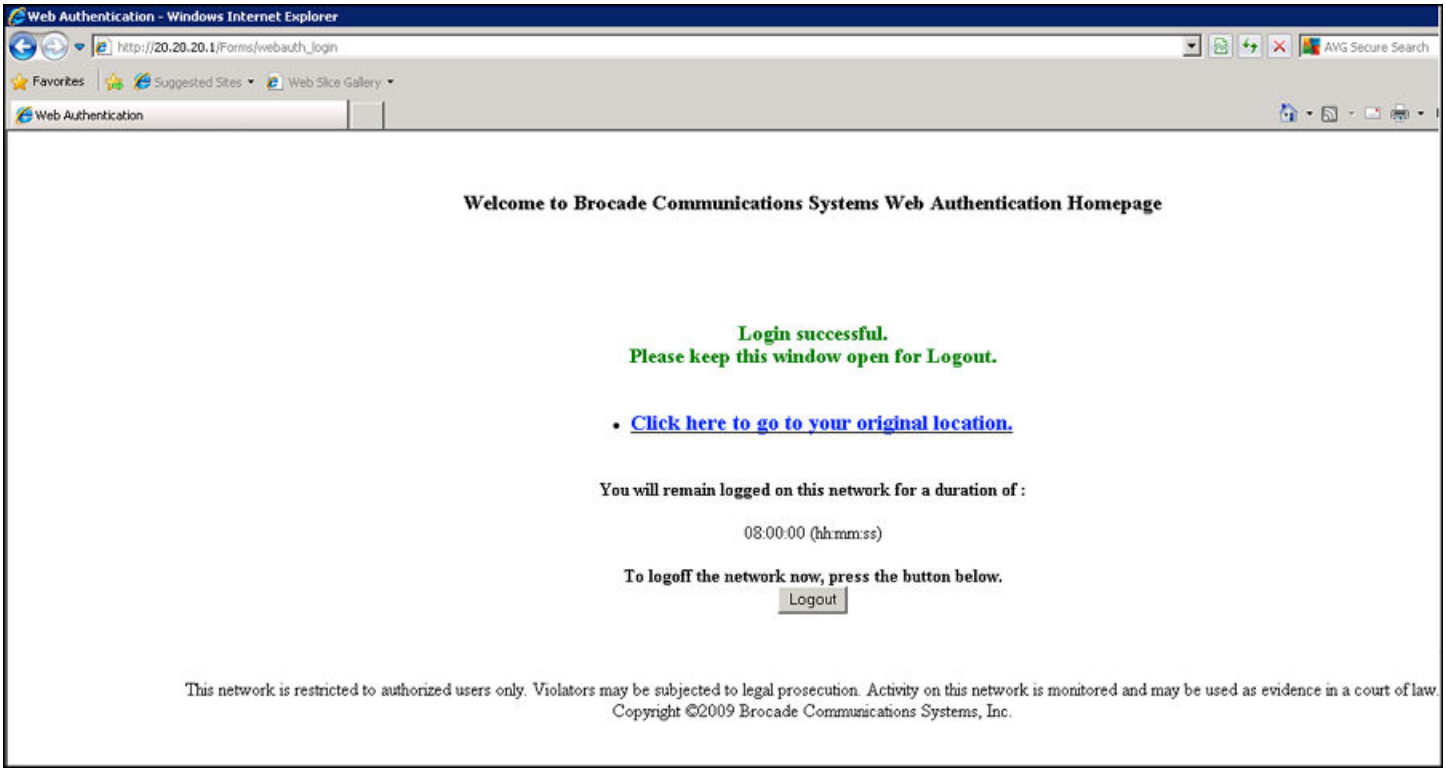
If the number of Web Authentication attempts by a user has been exceeded, the maximum attempts limit page is displayed. The user is blocked from attempting Web Authentication until either the user MAC address is removed from the blocked list (using the **clear webauth block-mac** command) or the block duration timer expires.

FIGURE 25 Example of a maximum attempts limit page



If Web Authentication is successful, the following success page appears.

FIGURE 26 Example of a Web Authentication success page

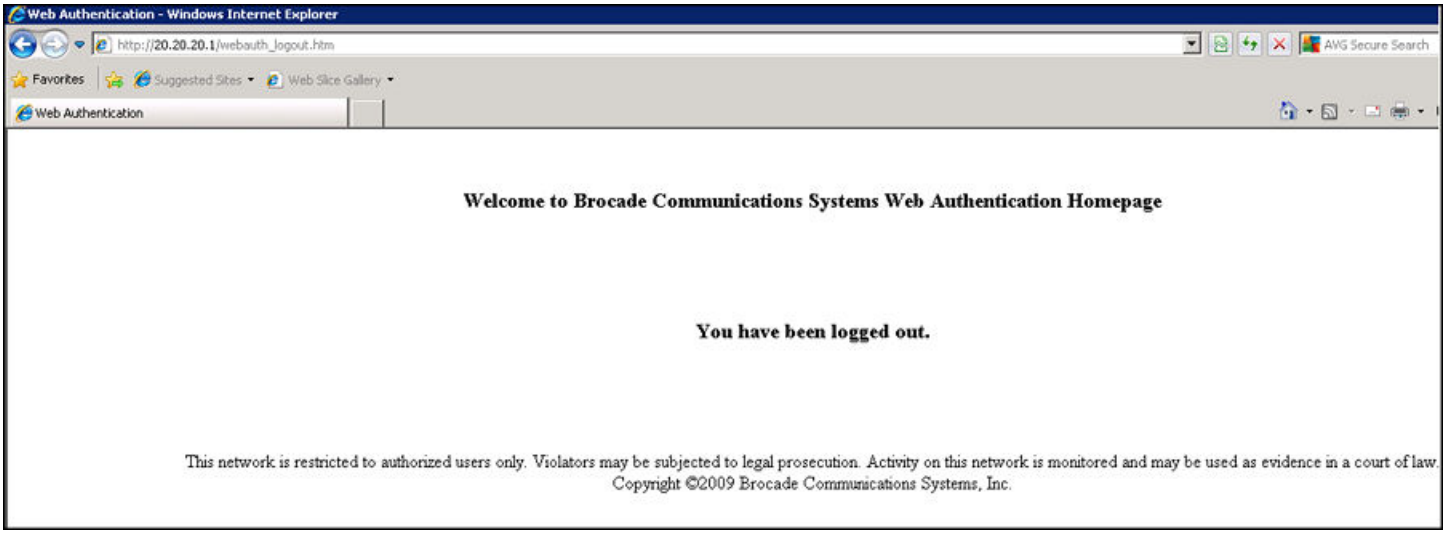


Once a host is authenticated, that host can manually de-authenticate by clicking the Logout button on the login success page. The host remains logged in until the re-authentication period expires. At that time, the host is automatically logged out. However, if a re-authentication period is not configured, 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 27 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 Brocade Communications, Inc. Web Authentication Homepage</h3>"
  Bottom (Footer): Custom Text
        "Copyright 2009 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

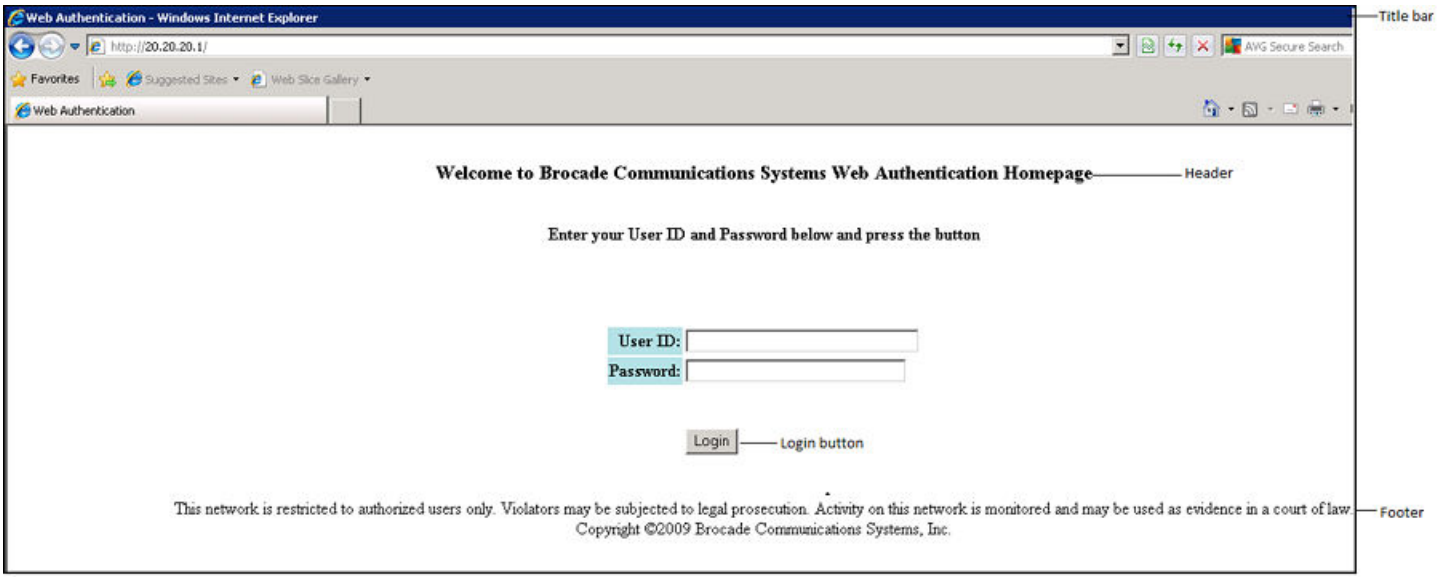
- Banner image (logo)
- Header
- Text box
- Login button
- Footer

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 28 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 "Brocade 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 brocadelogo.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"
```

Syntax: [no] webpage custom-text top *text*

You can enter up to 255 alphanumeric characters for the header. The default text is "Welcome to Brocade Communications, 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"
```

Syntax: [no] webpage custom-text bottom *text*

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 2009 Brocade Communications, Inc."

To reset the footer to the default text, enter the **no webpage custom-text bottom**.

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 34 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.
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 279.
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 285.
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    brocade     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    brocade     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 35 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 36 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.

TABLE 36 Field description of the show webauth **authenticating-list** command output (continued)

Field	Description
# 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 mode  Block Duration Remaining
MAC Address     User Name  mode   Static/Dynamic
--------------------------------------------------------------------------------
000c.2973.a42b  User1      E      D                   00:00:04
```

The **show webauth blocked-list** command displays the following information.

TABLE 37 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
```

Syntax: show webauth vlan *vlan-id* **passcode**

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-brocade
Configured Captive Portal Profile Details :
  cp-name               :cp-brocade
  virtual-ip            :10.21.240.42
  virtual-port          :80
  user-role             :guest
  login-page            :brocadeguestlogin.php
```

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.

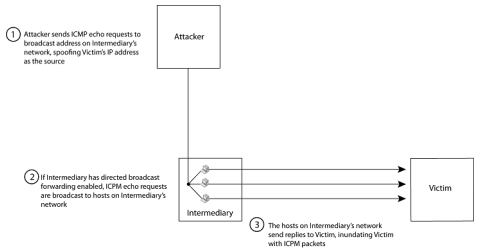
Brocade devices include measures to defend against two types of DoS attacks: Smurf attacks and TCP SYN attacks.

- 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 29 illustrates how a smurf attack works.

FIGURE 29 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
```

Syntax: [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.

For example, to set threshold values for ICMP packets targeted at the router, enter the following command in global CONFIG mode.

```
device(config)#ip icmp burst-normal 5000 burst-max 10000 lockup 300
```

For a ICX 7750 device, enter the following command in global CONFIG mode.

```
device(config)#ip icmp attack-rate burst-normal 2500 burst-max 3450 lockup 50
```

To set threshold values for ICMP packets received on interface 1/3/11, enter the following commands.

```
device(config)#interface ethernet 1/3/11
device(config-if-e1000-1/3/11)#ip icmp burst-normal 5000 burst-max 10000 lockup 300
```

To set threshold values for ICMP packets received on interface 1/3/11 for a ICX 7750 device, enter the following commands.

```
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 in the previous example. When ICMP 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 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 VE 31, enter commands such as the following.

```
device(config)#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 VE 31 for a ICX 7750 device, enter commands such as the following.

```
device(config)#interface ve 31
device(config-vif-31)#ip icmp attack-rate burst-normal 5000 burst-max 10000 lockup 300
```

Syntax: [no] ip icmp attack-rate burst-normal *value* **burst-max** *value* **lockup** *seconds*

The **attack-rate** parameter is specific to ICX 7750 and has no associated value.

The **burst-normal** *value* parameter can be from 1 through 100,000 packets per second.

The **burst-max** *value* paramter 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.

> **NOTE**
> For ICX 7750, the units of "burst-normal" and "burst-max" values are Kbps.

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.

In the example, if the number of ICMP packets received per second exceeds 5,000, 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 (5 minutes).

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.

For example, to set threshold values for TCP SYN packets targeted at the router, enter the following command in global CONFIG mode.

```
device(config)#ip tcp burst-normal 10 burst-max 100 lockup 300
```

To set threshold values for TCP SYN packets received on interface 1/3/11, enter the following commands.

```
device(config)#interface ethernet 1/3/11
device(config-if-e1000-1/3/11)#ip tcp burst-normal 10 burst-max 100 lockup 300
```

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. Otherwise, you can configure this feature at the interface level as shown in the previous example. WhenTCP/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.

NOTE
For ICX 7750 devices, the "attack rate" parameter is only applicable for smurf attacks and not for TCP/SYN attacks.

To set threshold values for TCP/SYN packets received on VE 31, enter commands such as the following.

```
device(config)#interface ve 31
device(config-vif-31)#ip tcp burst-normal 5000 burst-max 10000 lockup 300
```

Syntax: ip tcp burst-normal *value* **burst-max** *value* **lockup** *seconds*

NOTE
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.

The **burst-normal***value* parameter can be from 1 - 100,000 packets per second.

The **burst-max***value* parameter can be from 1 - 100,000 packets per second.

The **lockup***seconds* parameter can be from 1 - 10,000 seconds.

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.

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 (5 minutes).

TCP security enhancement

A TCP security enhancement improves the way TCP inbound segments are handled. This enhancement eliminates or minimizes the possibility of a TCP reset attack, in which a perpetrator attempts to prematurely terminate an active TCP session, and a data injection attack, where an attacker injects or manipulates data in a TCP connection.

In both cases, the attack is blind, meaning the perpetrator does not have visibility into the content of the data stream between two devices, but blindly injects traffic. The attacker also does not see the direct effect (the continuing communications between the devices and the impact of the injected packet) but may see the indirect impact of a terminated or corrupted session.

The TCP security enhancement prevents and protects against the following types of attacks:

- Blind TCP reset attack using the reset (RST) bit.
- Blind TCP reset attack using the synchronization (SYN) bit
- Blind TCP data injection attack

The TCP security enhancement is automatically enabled. If necessary, you can disable this feature. Refer to Disabling the TCP security enhancement on page 297.

Protecting against a blind TCP reset attack using the RST bit

In a blind TCP reset attack using the RST bit, a perpetrator attempts to guess the RST segments to prematurely terminate an active TCP session.

To prevent a user from using the RST bit to reset a TCP connection, the RST bit is subject to the following rules when receiving TCP segments:

- If the RST bit is set and the sequence number is outside the expected window, the device silently drops the segment.
- If the RST bit is exactly the next expected sequence number, the device resets the connection.
- If the RST bit is set and the sequence number does not exactly match the next expected sequence value, but is within the acceptable window, the device sends an acknowledgement (ACK).

The TCP security enhancement is enabled by default. To disable it, refer to Disabling the TCP security enhancement on page 297.

Protecting against a blind TCP reset attack using the SYN bit

For a blind TCP reset attack, the attacker tries to guess the SYN bits to terminate an active TCP session.To protect against this type of attack, the SYN bit is subject to the following rules during arrival of TCP segments:

- If the SYN bit is set and the sequence number is outside the expected window, the device sends an ACK to the peer.
- If the SYN bit is set and the sequence number is an exact match to the next expected sequence, the device sends an ACK segment to the peer. Before sending the ACK segment, the software subtracts a 1 from the value being acknowledged.
- If the SYN bit is set and the sequence number is acceptable, the device sends an ACK segment to the peer.

This TCP security enhancement is enabled by default. To disable it, refer to Disabling the TCP security enhancement on page 297.

Protecting against a blind injection attack

In a blind TCP injection attack, the attacker tries to inject or manipulate data in a TCP connection. To reduce the chances of a blind injection attack, an additional check is performed on all incoming TCP segments.

This TCP security enhancement is enabled by default. To disable it, refer to Disabling the TCP security enhancement on page 297.

Disabling the TCP security enhancement

Displaying statistics from a DoS attack

To display information about ICMP and TCP SYN packets dropped because burst thresholds were exceeded, 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
```

Syntax: show statistics dos-attack

To clear statistics about ICMP and TCP SYN packets dropped because burst thresholds were exceeded, enter the **clear statistics dos-attack** command.

```
device# clear statistics dos-attack
```

Syntax: clear statistics dos-attack

Clear DoS attack statistics

To clear statistics about ICMP and TCP SYN packets, enter the **clear statistics dos-attack** command.

```
device(config)# clear statistics dos-attack
```

Syntax: clear statistics dos-attack

```
---------------------------- Local Attack Statistics --------------------------
ICMP Drop Count    ICMP Block Count    SYN Drop Count    SYN Block Count
---------------    ----------------    --------------    ---------------
              0                   0                 0                  0
```

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.

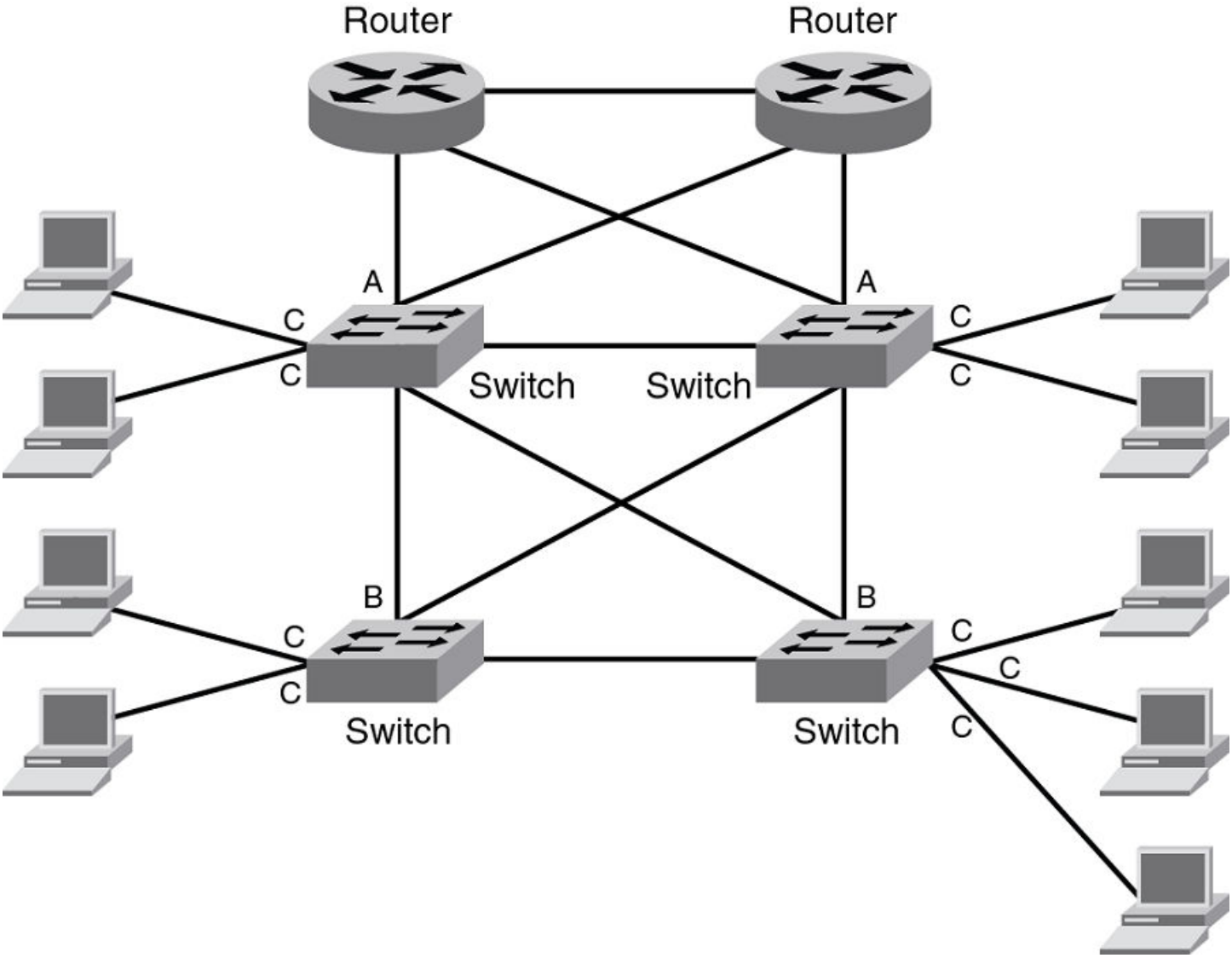
```
Brocade(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:1
Brocade(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:3
Brocade(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:10
Brocade(config)# ipv6 raguard policy policy1
Brocade(ipv6-RAG-policy policy1)# whitelist 1
Brocade(ipv6-RAG-policy policy1)# prefix-list raguard-prefix1
Brocade(ipv6-RAG-policy policy1)# preference-maximum medium
Brocade(ipv6-RAG-policy policy1)# logging
Brocade(ipv6-RAG-policy policy1)# exit
Brocade(config)# interface ethernet 1/1/1
Brocade(config-int-e1000-1/1/1)# raguard untrusted
Brocade(config-int-e1000-1/1/1)# exit
Brocade(config)# ipv6 raguard vlan 1 policy policy1
```

```
Brocade(config)# show ipv6 raguard all
Brocade(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 30 IPv6 RA guard configuration in a network



Configuring port A:

Configure port A as a trusted port.

```
Brocade(config)# interface ethernet 1/1/1
Brocade(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.

```
Brocade(config)# ipv6 raguard policy policyC
Brocade(ipv6-RAG-policy policyC)# exit
Brocade(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.

```
Brocade(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:10
Brocade(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:5
Brocade(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:12
Brocade(config)# prefix-list raguard-prefix-list1 permit 2001:db8::/16
Brocade(config)# ipv6 raguard policy policyB
Brocade(ipv6-RAG-policy policyB)# whitelist 1
Brocade(ipv6-RAG-policy policyB)# prefix-list raguard-prefix-list1
Brocade(ipv6-RAG-policy policyB)# exit
Brocade(config)# interface ethernet 1/1/2
Brocade(config-int-e1000-1/1/2)# raguard untrust
Brocade(config-int-e1000-1/1/2)# exit
Brocade(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.

```
Brocade# 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.

```
Brocade# 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 *Brocade 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.