CONFIGURATION GUIDE

RUCKUS FastIron Security Configuration Guide, 10.0.00

Supporting FastIron Software Release 10.0.00

Part Number: 53-1005753-10
Publication Date: 30 June 2023

© 2023 CommScope, Inc. All rights reserved.

No part of this content may be reproduced in any form or by any means or used to make any derivative work (such as translation, transformation, or adaptation) without written permission from CommScope, Inc. and/or its affiliates (“CommScope”). CommScope reserves the right to revise or change this content from time to time without obligation on the part of CommScope to provide notification of such revision or change.

Export Restrictions

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

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

Disclaimer

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

Limitation of Liability

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

Trademarks

CommScope and the CommScope logo are registered trademarks of CommScope and/or its affiliates in the U.S. and other countries. For additional trademark information see https://www.commscope.com/trademarks. All product names, trademarks, and registered trademarks are the property of their respective owners.

Patent Marking Notice

For applicable patents, see www.cs-pat.com.

Contents

Contact Information, Resources, and Conventions **11**
Contacting RUCKUS Customer Services and Support 11
What Support Do I Need? 11
Open a Case 11
Self-Service Resources 12
Document Feedback 12
RUCKUS Product Documentation Resources 12
Online Training Resources 12
Document Conventions 13
Notes, Cautions, and Safety Warnings 13
Command Syntax Conventions 13

About This Document **15**
New in This Document 15
Supported Hardware 16

Managing User Accounts **17**
User Accounts Overview 17
User Account Guidelines 17
Configuration Enhancements for Local User Accounts 18
Configuring Local User Accounts 20
Configuring Advanced Local User Account Features 21
Modifying Local User Account Passwords or Privileges 22
Deleting Local User Accounts 23
Enabling SSH Access 24
Recovering from a Lost Password 24

TACACS+ Server Authentication **25**
TACACS+ Security 25
How TACACS+ Differs from TACACS 25
TACACS+ Authentication, Authorization, and Accounting 25
TACACS+ Authentication 26
TACACS+ Authorization 26
TACACS+ Accounting 27
AAA Operations for TACACS+ 27
TACACS+ Configuration 28
TACACS+ Configuration Considerations 28
Identifying the TACACS+ Servers 28
Configuring Authentication-method Lists for TACACS+ 29
Specifying Different Servers for Individual AAA Functions 30
Setting Optional TACACS+ Parameters 30
Configuring TACACS+ Authorization 31
TACACS+ Accounting Configuration 34
Configuring an Interface as the Source for All TACACS+ Packets 35
Configuring TACACS+ for Devices in a Traditional Stack 35
TACACS+ Configuration Example 35
Displaying TACACS+ Statistics and Configuration Information 35

RADIUS Authentication **37**
RADIUS Security 37
RADIUS Authentication 37
RADIUS Authorization 38
RADIUS Accounting 38
AAA Operations for RADIUS 39
AAA Security for Commands Pasted into the running-config 39
RADIUS Configuration Considerations 40
Configuring RADIUS (Overview) 40
Configuring Company-Specific Attributes on the RADIUS Server 41
Identifying the RADIUS Server to the RUCKUS Device 42
Configuring an SSL Profile for Use with RADIUS Server Hosts 43
Specifying Different Servers for Individual AAA Functions 44
TLS and RADIUS 44
RADIUS Security Configuration Example 45
Mapping RADIUS Servers to Ports 46
RADIUS Configuration Example 47
Setting Up RADIUS over IPv6 47
Setting RADIUS Parameters 48
Configuring Detection of Dead RADIUS Servers 48
Source Address Configuration for RADIUS Packets 49
Configuring Authentication-method Lists for RADIUS 50
Authentication-Method Values 51
RADIUS Authorization 51
Configuring Exec Authorization 51
Configuring Command Authorization 51
Enabling RADIUS CoA and Disconnect Message Handling for Dynamic Authorization 52
RADIUS Accounting 54
Configuring RADIUS Accounting for Telnet/SSH (Shell) Access 54
Configuring RADIUS Accounting for CLI Commands 54
Configuring RADIUS Accounting for System Events 55
RADIUS Accounting for 802.1X Authentication and MAC Authentication 55
Displaying RADIUS Configuration Information 56

Authentication-Method Lists **59**
Authentication-Method List Overview 59
Configuration Considerations for Authentication-method Lists 59
Examples of Authentication-method Lists 60

ICX Digital Certificates **63**
Overview of ICX Digital Certificates 63
SSL Security 63
Enabling the SSL Server on the Device 64
Importing Digital Certificates and RSA Private Key Files 64
Displaying Information about ICX Digital Certificates 65
TLS Support 65

Secure Shell (SSH) **67**
SSH Version 2 Overview 67
Tested SSHv2 Clients 67
SSHv2 Supported Features 68
SSHv2 Unsupported Features 68

SSHv2 Authentication Types....68
Configuring SSHv2....69
Enabling and Disabling SSH by Generating and Deleting Host Keys....69
Configuring RSA Challenge-Response Authentication....71
Deleting the Public Keys....74
Optional Parameters Overview....74
SSH Rekey Configuration Notes....75
Setting Optional Parameters....75
Terminating an Active SSH Connection....76
SSHv2 Client....76
Enabling the SSHv2 Client....77
Configuring SSHv2 Client Public Key Authentication....77
Establishing an SSHv2 Client Connection....77
Enabling SSH Access....78
Displaying SSH Information....78

SCP Client Support....81
SCP Client....81
SCP Client Support Limitations....81
Supported SCP Client Configurations....82
Downloading an Image from an SCP Server....82
Uploading an Image to an SCP Server....82
Uploading Configuration Files to an SCP Server....83
Downloading Configuration Files from an SCP Server....83
Secure Copy with SSH2....83
Enabling and Disabling SCP....83
Secure Copy Configuration Notes....84

ACLs....85
Layer 3 ACL Overview....85
ACL Scaling....86
Default ACL Action....88
How Hardware-based ACLs Work....89
How Fragmented Packets Are Processed....89
IPv4 ACLs....89
IPv4 ACL Configuration Guidelines....90
Creating and Applying a Standard IPv4 ACL....91
IPv4 Extended ACL Traffic Filtering Criteria....92
Creating and Applying an Extended IPv4 ACL....92
Applying Egress ACLs to Control (CPU) Traffic....93
Enabling Strict Control of ACL Filtering of Fragmented Packets....93
Filtering on IP Precedence and ToS Values....93
ACLs to Filter ARP Packets....94
QoS Options for IP ACLs....96
ACL-based Rate Limiting....99
ACLs to Control Multicast Features....99
Displaying IPv4 ACL Information....99
IPv6 ACLs....100
IPv6 ACL Traffic Filtering Criteria....100
IPv6 Protocol Names and Numbers....101
Default and Implicit IPv6 ACL Action....101

IPv6 ACL Configuration Notes ... 102
Creating and Applying an IPv6 ACL ... 103
Neighbor Discovery (ND)-Packet DoS Attacks ... 104
Displaying IPv6 ACLs ... 105
Applying an ACL to a LAG Interface ... 107
Applying ACLs to VLANs ... 107
ACL Logging ... 108
Configuration Notes for ACL Logging ... 108
Enabling ACL Logging ... 109
ACL Statistics ... 110
ACL Accounting ... 111
Limitations for ACL Accounting ... 111
Changing the Accounting Period ... 111
Configuring ACL Accounting ... 112
Adding a Comment for an Entry in an ACL ... 114
Deleting a Comment from an ACL Entry ... 115
Viewing Comments in an ACL ... 115
Sequence-based ACL Editing ... 116
Inserting Rules into ACLs ... 116
Deleting Rules from ACLs ... 117
Displaying TCAM Information for ACLs ... 118

MAC ACLs ... 125
Layer 2 ACL Overview ... 125
MAC Scaling by ICX Device ... 125
MAC ACL Default Action ... 126
MAC ACL Configuration Notes and Limitations ... 126
Configuring and Applying MAC ACLs ... 126
Displaying MAC ACL Information ... 128

Policy-Based Routing ... 131
Policy-Based Routing Overview ... 131
PBR Permit and Deny Actions ... 131
LAG Formation with PBR Policy ... 132
Route Maps ... 132
Match Statement ... 132
Set Statement ... 132
Configuration Guidelines for IPv4 PBR ... 133
Configuring an IPv4 PBR Policy with an IPv4 Address as the Next Hop ... 134
Configuring an IPv4 PBR Policy with the NULL0 Interface as the Next Hop ... 135
Configuring an IPv4 PBR Policy with a Tunnel as the Next Hop ... 136
Configuring an IPv4 PBR Policy by Setting a VRF-aware Next Hop in a Route Map ... 137
Displaying IPv4 PBR Information ... 138
Configuration Guidelines for IPv6 PBR ... 138
Configuring an IPv6 PBR Policy with an IPv6 Address as the Next Hop ... 139
Configuring an IPv6 PBR Policy with the NULL0 Interface as the Next Hop ... 140
Configuring an IPv6 PBR Policy with a Tunnel as the Next Hop ... 141
Displaying IPv6 PBR Information ... 142

Media Access Control Security ... 143
MACsec Overview ... 143
Supported MACsec Hardware Configurations ... 143

MACsec RFCs and Standards ... 143
MACsec Considerations ... 144
How MACsec Works ... 144
How MACsec Handles Data and Control Traffic ... 144
MACsec Key Agreement Protocol ... 144
MKA Message Exchange between Two Switches ... 145
Secure Channels ... 145
MACsec Frame Format ... 145
Configuring MACsec ... 148
Enabling MACsec and Configuring Group Parameters ... 149
Configuring MACsec Key-Server Priority ... 149
Configuring MACsec Integrity and Encryption ... 150
Configuring MACsec Frame Validation ... 151
Configuring Replay Protection ... 151
Configuring Data-Delay Protection ... 152
Enabling and Configuring Group Interfaces for MACsec ... 153
Configuring the Pre-shared Key ... 153
Sample MACsec Configuration ... 154
Displaying MACsec Information ... 154
Displaying MACsec Configuration Details ... 155
Displaying Information on Current MACsec Sessions ... 156
Displaying MKA Protocol Statistics for an Interface ... 157
Displaying MACsec Secure Channel Activity for an Interface ... 158

Port MAC Security (PMS) ... 161
Port MAC Security Overview ... 161
Local and Global Resources Used for Port MAC Security ... 161
Configuration Considerations for Port MAC Security ... 162
Secure MAC Movement ... 162
Port MAC Security Configuration ... 162
Configuring Port MAC Security ... 163
Displaying Port MAC Security Information ... 164
Clearing Restricted MAC Addresses and Port Security Violation Statistics ... 165

Flexible Authentication ... 167
Flexible Authentication Overview ... 167
MAC VLANs ... 168
Data VLAN Requirements for Flexible Authentication ... 168
Voice VLAN Requirements for Flexible Authentication ... 169
Authentication Modes ... 170
Tagged VM Client Support ... 170
Static Authentication with MAC Authentication Filters ... 171
Authentication Actions ... 171
Authentication Session Limits on an Interface ... 172
How Flexible Authentication Works ... 173
Configuration Considerations and Guidelines for Flexible Authentication ... 176
802.1X Authentication ... 177
Device Roles in an 802.1X Configuration ... 177
Communication Between the Devices ... 178
Controlled and Uncontrolled Ports ... 179
Port Control for Authentication ... 179

Message Exchange During Authentication....180
MAC Authentication....181
MAC Address Formats Sent to the RADIUS Server....182
Authenticating Multiple Hosts Connected to the Same Port....182
How Flexible Authentication Works for Multiple Clients....183
Flexible Authentication Accounting....183
Change of Authorization....184
Multiple RADIUS Servers....184
Session Aging....185
Periodic Reauthentication of Authenticated Clients....185
Denial of Service Protection Support....186
SNMP Traps for Flexible Authentication....186
Syslog Messages for Flexible Authentication....186
RADIUS Attributes for Authentication and Accounting....186
Configuring ICX Vendor-Specific Attributes on the RADIUS Server....189
Support for the RADIUS User-Name Attribute in Access-Accept Messages....191
Dynamic VLAN Assignment....191
Configuring the RADIUS Server to Support Dynamic VLAN Assignment for Authentication....192
Authentication Success Scenarios....193
Authentication Failure Scenarios....194
Authentication Server Timeout Scenarios....195
Authentication Client Timeout Scenarios (No Response to EAP Packets)....196
Automatic Removal of Dynamic VLAN Assignments for 802.1X and MAC Authenticated Ports....196
Dynamic ACLs in Authentication....196
ACL Filters Downloaded from RADIUS....197
Considerations for Downloadable ACLs Configured on a RADIUS Server....197
Dynamically Applying Existing ACLs in Flexible Authentication....198
Support for IP Source Guard Protection....199
Configuring Flexible Authentication....200
Flexible Authentication Configuration Prerequisites....200
Configuring Flexible Authentication Globally....201
Configuring Flexible Authentication on an Interface....203
Enabling 802.1X Authentication....205
Enabling MAC Authentication....206
Excluding the RADIUS Server for Login Features....208
Displaying Authentication Information....208
Displaying Configuration....208
Displaying Statistics....209
Displaying the Authentication Sessions....210
Displaying Information About User ACLs....212
Displaying Dynamically Assigned VLAN Information....212
Clearing Authentication Details....212

HTTP and HTTPS....215
Web Authentication Overview....215
Captive Portal Authentication (External Web Authentication)....216
Captive Portal Profile for External Web Authentication....217
Captive Portal on a VLAN....217
Dynamic IP ACLs in Web Authentication....218
Configuration Considerations for Applying IP ACLs....219
Dynamically Applying Existing ACLs (HTTP and HTTPS)....219

RADIUS Attribute for Session Timeout ... 220
Web Authentication Configuration Considerations ... 220
Configuring Web Authentication ... 221
Prerequisites for Captive Portal Support with RUCKUS Cloudpath ... 222
Prerequisites for Configuring Captive Portal with Aruba ClearPass ... 224
Prerequisites for Configuring External Web Authentication with Cisco ISE ... 226
Prerequisite Configurations on an ICX Switch for Captive Portal Authentication ... 227
Creating the Captive Portal Profile for External Web Authentication ... 227
Configuring Captive Portal (External Web Authentication) ... 228
Web Authentication Mode Configuration ... 230
Using Local User Databases ... 231
Passcodes for User Authentication ... 232
Automatic Authentication ... 235
Web Authentication Options ... 235
RADIUS Accounting for Web Authentication ... 236
Trusted Ports ... 236
Permanently Authenticated Hosts ... 236
IP Addresses and Domain Names Allowed During Web Authentication ... 236
Re-authentication Period ... 237
Web Authentication Cycle ... 237
Limiting the Number of Web Authentication Attempts ... 237
Clearing Authenticated Hosts from the Web Authentication Table ... 237
Block Duration for Web Authentication Attempts ... 237
Manually Blocking a Specific Host ... 237
Limiting the Number of Authenticated Hosts ... 237
Filtering DNS Queries ... 237
Forcing Re-authentication When Ports Are Down ... 238
Forcing Re-authentication After an Inactive Period ... 238
Defining the Web Authorization Redirect Address ... 238
Customizing the Web Authentication Redirect Page ... 238
Honoring the RADIUS-Returned VLAN ... 238
Configuring Web Authentication Options ... 239
Web Authentication Pages ... 240
Displaying Web Authentication Information ... 247
Image Download over HTTPS ... 250
Considerations for Downloading Images over HTTPS ... 250
Configuration Download over HTTPS ... 251
Considerations for Downloading a Configuration File over HTTPS ... 251
Configuration Upload over HTTPS ... 251
Considerations for Uploading a Configuration File over HTTPS ... 252

Protecting against Denial of Service Attacks ... 253
Denial of Service Protection Overview ... 253
Protecting against Smurf Attacks ... 253
Avoiding Being an Intermediary in a Smurf Attack ... 254
Avoiding Being a Victim in a Smurf Attack ... 254
Protecting against TCP SYN Attacks ... 255
Configuring Threshold Values for TCP SYN Packets Globally ... 256
Configuring TCP SYN Threshold Values on an Interface ... 256
TCP MSS Adjustment Overview ... 257
Example ... 257

Impact on Existing Functionality ... 258
TCP MSS Adjustment Limitations ... 258
Defense against TCP Denial of Service Attacks on ICX 8200 Devices ... 258
Rate Limiting TCP Control Traffic ... 258
Dropping Irregular TCP Control Packets ... 259
Defense against ICMP Denial of Service Attacks on ICX 8200 Switches ... 259
Rate Limiting ICMP Control Traffic ... 259
Dropping Suspicious ICMP Packet Fragments ... 260
Displaying Statistics from a DoS Attack ... 260
DOS Show Commands for ICX 8200 Devices ... 260
Clear DoS Attack Statistics ... 261

IPv6 RA Guard ... 263
Securing IPv6 Address Configuration ... 263
IPv6 RA Guard Overview ... 263
RA Guard Policy ... 263
Whitelist ... 263
Prefix List ... 264
Maximum Preference ... 264
Trusted, Untrusted, and Host Ports ... 264
Configuration Notes and Feature Limitations for IPv6 RA Guard ... 264
Configuring IPv6 RA Guard ... 264
Example of Configuring IPv6 RA Guard ... 265
Example: Configuring IPv6 RA Guard on a Device ... 265
Example: Configuring IPv6 RA Guard in a Network ... 265
Example: Verifying the RA Guard Configuration ... 267

Joint Interoperability Test Command ... 271
JITC Overview ... 271
AES-CTR Encryption Mode Support for SSH ... 271
SHA1 Authentication Support for NTP ... 271
IPv6 ACL for SNMPv3 Group ... 271

OpenSSL License ... 273
OpenSSL License ... 273
Original SSLeay License ... 273

Keychain Module ... 275
Keychain Module Overview ... 275
How the Keychain Module Works ... 275
Components of a Keychain ... 275
OSPF Keychain Authentication ... 276
Configuring a Keychain Module ... 276
TCP Keychain Options ... 277
TCP Keychain Considerations ... 278
Configuring TCP Keychain Options ... 278
MKA Keychain Overview and Considerations ... 280
Creating and Configuring an MKA Keychain ... 281

Contact Information, Resources, and Conventions

- Contacting RUCKUS Customer Services and Support 11
- Document Feedback 12
- RUCKUS Product Documentation Resources 12
- Online Training Resources 12
- Document Conventions 13
- Command Syntax Conventions 13

Contacting RUCKUS Customer Services and Support

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

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

What Support Do I Need?

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

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

Open a Case

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

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

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

Self-Service Resources

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

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

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

Document Feedback

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

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

When contacting us, include the following information:

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

For example:

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

RUCKUS Product Documentation Resources

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

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

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

Online Training Resources

To access a variety of online RUCKUS training modules, including free introductory courses to wireless networking essentials, site surveys, and products, visit the RUCKUS Training Portal at https://commscopeuniversity.myabsorb.com/. The registration is a two-step process described in this video. You create a CommScope account and then register for, and request access for, CommScope University.

Document Conventions

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

TABLE 1 Text Conventions

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

Notes, Cautions, and Safety Warnings

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

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

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



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



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

Command Syntax Conventions

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

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

About This Document

- New in This Document 15
- Supported Hardware..... 16

New in This Document

The following table describes changes to this guide for the FastIron 10.0.00 software release.

TABLE 2 Summary of Changes in FastIron Release 10.0.00a

Feature	Description	Reference
Strict password enforcement	**New**:The **enable strict-password-enforcement** command is reinstated in this release.	Configuring Advanced Local User Account Features on page 21
Updates to address defects	**Updated**: Minor updates have been introduced throughout the guide to address defects.	All chapters
Minor editorial updates	**Updated**: Minor editorial updates were made throughout the guide.	All chapters

TABLE 3 Summary of Changes in FastIron Release 10.0.00

Feature	Description	Reference
RUCKUS ICX 8200	**New**: Support for RUCKUS ICX 8200 devices has been added.	Throughout the guide
Discontinuation of support for a separate switch image	**Updated**: This release and future releases support a single image for ICX platforms that provides both switching and routing functionality. As a result, support for a separate switch ("Layer 2") image is discontinued.	References have been removed throughout the guide.
Defense against Denial of Service Attacks (DDOS) Enhancements	**New**: New commands are available on ICX 8200 switches for rate limiting TCP SYN and ICMP control traffic to prevent Denial of Service (DOS) attacks. Several commands have also been introduced on ICX 8200 switches to drop irregular, suspect TCP SYN and ICMP control packets to prevent DOS attacks.	Defense against TCP Denial of Service Attacks on ICX 8200 Devices on page 258 Defense against ICMP Denial of Service Attacks on ICX 8200 Switches on page 259
RADIUS-Downloadable ACLs	**New**: User profiles configured with ACLs on a RADIUS server can be downloaded to an ICX device using Access-accept message attributes containing ACL filter statements.	Dynamic ACLs in Authentication on page 196
Expanded Maximum VLANs in Tunnel-Private-Group-ID	**Updated**: The maximum number of Tunnel-Private-Group-IDs has increased from 8 VLAN IDs to 32.	Configuring the RADIUS Server to Support Dynamic VLAN Assignment for Authentication on page 192
ICX 8200 Behavior with IPv6 ACL Packet Fragments	**New**: ICX 8200 switches include the first IPv6 packet fragments in access control list (ACL) rule matching. In contrast, other ICX devices include non-first fragments in rule matching.	IPv6 ACL Traffic Filtering Criteria on page 100
ICX 8200 ACL Scaling	**Updated**: Scaling numbers for ICX 8200 switches have been included in ICX device tables.	ACL Scaling on page 86
ICX 8200 Behavior in Flexible Authentication	**New**: In Flexible authentication, ICX 8200 switches require a Layer 2 ingress ACL per user in multiple-untagged mode. The user will be blocked if there is not a matching ACL to apply in this mode.	Configuration Considerations and Guidelines for Flexible Authentication on page 176

TABLE 3 Summary of Changes in FastIron Release 10.0.00 (continued)

Feature	Description	Reference
Unsupported hardware	**Updated**: This release and future releases do not support RUCKUS ICX 7150, 7250, or 7450 devices.	References have been removed throughout the guide.
Updates to address defects	**Updated**: Minor updates have been introduced throughout the guide to address defects.	All chapters
Minor editorial updates	**Updated**: Minor editorial updates were made throughout the guide.	All chapters

Supported Hardware

This guide supports the following RUCKUS products:

- RUCKUS ICX 8200 Switches
- RUCKUS ICX 7850 Switches
- RUCKUS ICX 7650 Switches
- RUCKUS ICX 7550 Switches

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

Managing User Accounts

- User Accounts Overview 17
- Configuring Local User Accounts 20
- Enabling SSH Access 24
- Recovering from a Lost Password 24

User Accounts Overview

Up to 32 local user accounts can be defined on a RUCKUS device. User accounts regulate who can access the management functions in the CLI.

You can create accounts for local users with passwords. Accounts passwords are encrypted. You can assign privilege levels to local user accounts.

NOTE
An ICX switch new from the factory includes a local account with Super-User privileges. You must retain a local account with Super-User privileges to create new users or set user privilege levels.

NOTE
FastIron 09.0.10a and later releases support only SHA-512 encryption for local user accounts. The following encryption types are deprecated:

- service password-encryption sha1
- service password-encrypiton sha256

Any user account created with one of the deprecated encryption methods is deleted when the ICX switch is upgraded to FastIron release 09.0.10a or later.

User accounts regulate access to the management functions in the CLI using the following methods:

- Telnet access
- Web management access
- SNMP access
- RESTCONF
- SSH access–SSH refers to SSHv2, which is supported on all RUCKUS ICX devices.

For each local user account, you specify a user name and password. You also can specify the management privilege level, which can be one of the following:

- 0 - 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.
- 4 - Port Configuration level - Allows read-and-write access for specific ports, but not for global parameters.
- 5 - Read Only level - Allows access to the Privileged EXEC mode and User EXEC mode with read access only.

User Account Guidelines

Be aware of the following guidelines for user accounts.

- If the message of the day (MOTD) is configured, it will be displayed after the Telnet, SSH, or console session is successfully authenticated. The MOTD is configured using the **banner motd** command.

- 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.

Configuration Enhancements for Local User Accounts

Advanced features are available when configuring users and their passwords.

Although basic local user accounts can be set up without many rules, most companies now require more security for user accounts. The following enhancements are disabled by default:

- Enhanced user password combination requirements
- User password masking
- Quarterly updates of user passwords
- Storage of up to 15 previously configured passwords for each user
- Configuration of the number of login attempts before users are locked out
- Password expiration

Password Combination Rules

You must enter a minimum of eight characters containing the following combinations when you create and enable a user password:

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

NOTE
Password minimum and combination requirements are strictly enforced.

The following security upgrades apply to strict password enforcement:

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

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

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

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

If the user tries to configure a password that was previously used, the new password configuration fails, and the following message is displayed.

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

Password Aging

For enhanced security, password aging enforces quarterly updates of all user passwords. After 90 days, the CLI automatically prompts users to change their passwords when they attempt to sign on.

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

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

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

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

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

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

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

Password History

By default, a 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 device to store up to 15 passwords for each user, so that users do not use the same password multiple times. The default number of stored passwords is 5. If a user attempts to use a stored password, the system prompts the user to choose a different password.

User Login Attempts

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. 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 by specifying a 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 and can be configured in the range of 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, you can perform one of the following actions:

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

> **NOTE**
> You must have user privilege level 0 to re-enable other users with this command.

Password Expiration

You can set a user password to expire. Once a password expires, the administrator must assign a new password to the user. The days before expiration can be set as a value from 1 through 365. The default is 90 days.

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

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

Configuring Local User Accounts

You must be logged in with Super User access (privilege level 0) to add or delete user accounts or configure or modify other access parameters. In the following task, various level of access are configured for users. After Step 2 is performed for a new device, all the steps are optional.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a user with the desired privilege level. On the same line, create a password.
   - privilege 0 - Super-user (default) allows complete read-and-write access to the system.
   - privilege 4 - Read-and-write access for specific ports (but not for global parameters)
   - privilege 5 - Read-only access.

     NOTE
     In FastIron 09.0.10a and later releases, both of the following command options create encrypted passwords (previously, the password option created an unencrypted password).

   - password
   - create-password

     ```
     device(config)# username super privilege 0 create-password xpassx
     device(config)#
     ```

3. To prevent unauthorized user account deletion enter the **service local-user-protection** command.

   ```
   device(config)# service local-user-protection
   ```

   The **service local-user-protection** command applies to all user accounts. When you try to delete a specific user you will be prompted for verification before deleting that user.

4. (Optional) Display user account information using the **show users** command.

The following example creates a user account with read/write access and an encrypted password using the **create-password** command.

```
device(config)# username user-mktg3 privilege 0 create-password xpassx
```

The following example creates a user with read-only privileges.

```
ICX7550-24P Router(config)# username read_user privilege 5 pass
  password          Specify the password for the user
ICX7550-24P Router(config)# username read_user privilege 5 password read_user
ICX7550-24P Router(config)# show user
Username                                        Password                          Encrypt   Priv Status
Expire Time
==============================================================================================================
==========
super                                           $1$M78fhauw$Pk4tQyYlhe722GG7WG2wZ  enabled   0    enabled
Never
read_user                                       $1$M78fhauw$8F.GCXRWk3mAVW/OxA3HB  enabled   5    enabled
Never
```

Configuring Advanced Local User Account Features

Advanced features providing more control and security are available when configuring user accounts and their passwords.

The following features are configured in this task. All these features are disabled by default:

- Password Length
- Password Combination Rules
- Password Aging
- Password History
- User Login Attempts
- Password Expiration

All the steps are optional and can be entered in any order.

NOTE
A factory-default super user account is present on new devices. On first log-in, the password for the factory-default account must be changed. See the *Configuring Local User Accounts* task for more details on adding local users.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable password aging to force the user to provide a new password every three months.

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

   After 90 days the CLI automatically prompts the user for a new password.

3. Enable a minimum number of characters and a required combination of characters to ensure secure passwords.

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

   When strict password enforcement is enabled, the password must be a minimum of 15 characters and must contain the following combinations:

   - At least two uppercase characters
   - At least two lowercase characters
   - At least two numeric characters
   - At least two special characters

   The strict password enforcement feature displays an error message when the password entered does not meet the criteria.

   NOTE
   Strict password enforcement is configured globally. Only accounts and passwords configured after the feature is enabled are subject to the minimum password length requirement.

4. Configure the device to store up to 15 previous passwords to prevent previous passwords from being used as a security measure.

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

   An error message will display if a user attempts to use a previous password that is still stored.

5. Configure the maximum number of invalid login attempts a user can make before being locked out to 8 with a 15 minute time period before the user account is automatically unlocked.

   ```
   device(config)# enable user disable-on-login-failure 8 login-recovery-time 15
   ```

   If the **login-recovery-time** option is not configured, manual intervention by an administrator is required to unlock the user account.
6. Configure a user password to expire in 30 days.

   ```
   device(config)# username sandy expires 30
   ```

   Password expiration can be used for temporary user accounts.
7. (Optional) Display user account information using the **show users** command.

The following example shows how to configure advanced local user account features to provide more secure user accounts and passwords, including password requirements imposed by the **enable strict-password-enforcement** command.

```
device# configure terminal
device(config)# enable strict-password-enforcement
device(config)# enable user password-aging
device(config)# enable user password-history 15
device(config)# enable user disable-on-login-failure 8 login-recovery-time 15
device(config)# username sandy expires 30
```

Modifying Local User Account Passwords or Privileges

You must be logged in with Super User access to modify user accounts. Some user accounts require authentication before modification.

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 the correct password is provided.

By default, a local user account can be modified without any authentication.

In the following task, user accounts are modified to change a password or a privilege level. You can modify both password and privilege level in the same step.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Change the password for a user account with local user protection service. You are prompted for the current password after confirmation.

   ```
   device(config)# username user-mktg3 create-password newxpassx
   User already exists, Do you want to modify: (enter 'y' or 'n') y
   To modify or remove user, enter current password: ******
   ```

   The new password can be up to 48 characters long and must be different from the current and previous two passwords.
3. Change the privilege level for a user account without local user protection service.

   ```
   device(config)# username user-mktg4 privilege 4
   ```

   In the example, user account user-mktg4 now has a privilege level of 4, which allows port configuration.
4. Exit global configuration mode.

   ```
   device(config)# exit
   ```

5. (Optional) To verify that you have modified the user accounts, display user account information using the **show users** command in Privileged EXEC mode.

Deleting Local User Accounts

You must be logged in with Super User access to delete user accounts. Some user accounts require authentication before deletion.

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 the correct password is provided.

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

NOTE
You will not be able to remove the last user account if the authentication method for login or web-server is "local." Deletion of the last available local user is allowed only if login and web-server authentication methods are tacacs+ or radius.

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

- AAA configuration specific to local authentication is removed.
- Unless at least one local user is present, the method "local" is unavailable in the AAA authentication-method list configuration. Refer to Authentication-Method Lists on page 59 for more information on the authentication-method list.

In the following task, user accounts with various levels of authentication are deleted. All the steps are optional.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Delete a user account with no password protection.

```
device(config)# no username user-mktg4
```

   By default, a local user account can be deleted without any authentication.

3. Delete a user account with encrypted password protection and local user protection service. You are prompted for the current password after confirmation.

```
device(config)# no username user-mktg3
User already exists, Do you want to modify: (enter 'y' or 'n') y
To modify or remove user, enter current password: ******
```

   NOTE
   The **service user-local-protection** command enforces verification for any changes to the user account.

4. Exit to Privileged EXEC mode.

```
device(config)# exit
```

5. To verify that you have deleted the user accounts, display user account information using the **show users** command.

```
device# show users
```

Enabling SSH Access

By default, there are no restrictions on access for management protocols, including SSH. To allow SSHv2 access to a RUCKUS ICX device, you must generate a Crypto Key.

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

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **crypto key generate** command to generate the crypto key to enable SSH.

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

The following example generates a client RSA key pair with a modulus size of 2048 bits.

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

The following example generates an elliptical key pair named testkey with the default size of 384 bits.

```
device# configure terminal
device(config)# crypto key generate ec label testkey
```

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. Reload the device.

```
device# reload
```

3. At the initial boot prompt at system startup, enter **b** to enter the boot monitor mode.

```
device> b
```

4. Enter the **no password** command at the prompt. (You cannot abbreviate this command.) This command will cause the device to bypass the system password check.

```
device-Boot> no password
```

5. Enter the **boot_primary** command at the prompt.

```
device-Boot> boot_primary
```

6. After the console prompt reappears, assign a new password.

TACACS+ Server Authentication

- TACACS+ Security....25
- TACACS+ Authentication, Authorization, and Accounting....25
- TACACS+ Configuration....28
- Displaying TACACS+ Statistics and Configuration Information....35

TACACS+ Security

The Terminal Access Controller Access Control System protocol TACACS+ defines how authentication, authorization, and accounting information is sent between a RUCKUS device and an authentication database on aTACACS+ server. TACACS+ services are maintained in a database, typically on a UNIX workstation or PC with a TACACS+ server running.

You can use the TACACS+ protocol 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

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 that uses TCP to ensure reliable delivery.

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 means any authentication mechanism can be used 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, and no special implementation or configuration is required to employ all three.

TACACS+ Authentication, Authorization, and Accounting

When you configure a RUCKUS device to use a TACACS+ server for authentication, the device prompts users who are trying to access the CLI for a user name and password and then verifies the password with the 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. As an option, you can also 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 first enters the User EXEC level and can then enter the **enable** command to access the Privileged EXEC level. A user who is successfully authenticated can be automatically placed at the Privileged EXEC level after login.

TACACS+ Authentication

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

NOTE
Multiple challenges are supported for TACACS+ login authentication.

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 a user is authenticated.
- Command authorization consults a TACACS+ server to obtain 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. The RUCKUS device looks at its configuration to see if the command is at a privilege level that requires TACACS+ command authorization.
3. 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.

4. 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 occurs on the RUCKUS device:
   - A user logs into the management interface using Telnet or SSH.
   - A user enters a command for which accounting has been configured.
   - A system event occurs, such as a reboot or reloading of the configuration file.
2. The RUCKUS device checks 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 containing information about the event to the TACACS+ accounting server.
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+

The following table lists the sequence of user actions and the related authentication, authorization, and accounting operations that take place when a user gains access to a RUCKUS device that has TACACS+ security configured. The operations column in the following table also includes the related commands to be configured on the RUCKUS device.

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* **defaultstart-stop** *method-list*
	EXEC accounting stop (TACACS+): **aaa accounting exec default start-stop** *method-list*
User enters system commands (for example, **reload**, **boot system**)	Command authorization (TACACS+): **aaa authorization commands** *privilege-level* **default** *method-list*
	Command accounting (TACACS+): **aaa accounting commands** *privilege-level* **default start-stop** *method-list*
	System accounting stop (TACACS+): **aaa accounting system default start-stop** *method-list*

User Action	Applicable AAA Operations
User enters the command: **[no] aaa accounting systemdefaultstart-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+ Configuration

Configuring TACACS+ authentication involves the following steps.

1. Identify one or more TACACS+ servers.
2. Configure authentication.
3. (Optional) Assign different functions to different servers if desired.
4. (Optional) Set optional TACACS+ parameters.
5. (Recommended, optional) Configure authorization.
   - Configure privilege-level authorization.
   - Configure command authorization.
6. (Optional) Configure accounting.

TACACS+ Configuration Considerations

- You must deploy at least one TACACS+ server in your network.
- RUCKUS devices support authentication using up to eight 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 login, CLI Privileged EXEC and CONFIG levels). For example, you can select TACACS+ as the primary authentication method for 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. Refer to Configuring Authentication-method Lists for TACACS+ on page 29 for more information.

Identifying the TACACS+ Servers

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

To identify a TACACS+ server, enter the **tacacs-server host** command followed by the address or name of the server as shown in the following example, which identifies three TACACS+ servers.

```
device# configure terminal
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
```

An IP address, IPv6 address, or hostname can be used to designate 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** command as shown in the following example.

```
device# configure terminal
device(config)# ip dns server-address 10.94.6.161
device(config)# tacacs-server host frog
```

If you add multiple TACACS+ authentication servers to the RUCKUS device, the device tries to reach them in the order you add them. For the previous example, the software tries the servers in the following order.

- 10.94.6.161
- 10.94.6.191
- 10.94.6.122

You can remove a 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 remove a TACACS+ server with the **no tacacs-server** command, make sure you also erase any related **aaa** commands that specify TACACS+ as an authentication method. Otherwise, when you exit from configuration mode or from a Telnet session, the system continues to try TACACS+ authentication, and you will not be able to access the system.

Configuring Authentication-method Lists for TACACS+

You can use TACACS+ to authenticate Telnet or SSH access as well as access to the Privileged EXEC level and CONFIG levels of the CLI. When configuring TACACS+ authentication, you create authentication-method lists specifically for the different types of access.

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

There are two different authentication-method lists for TACACS+ authentication.

- **Telnet/SSH** - Use the **aaa authentication login default** command followed by appropriate methods to create an authentication-method list for Telnet or SSH CLI access.
- **CLI** - Use the **aaa authentication enable default** command followed by appropriate methods to create a separate authentication-method list for access to the Privileged EXEC and CONFIG levels of the CLI.

The following example creates an authentication-method list that specifies TACACS+ as the primary authentication method for access to the Privileged EXEC level and CONFIG levels of the CLI. If TACACS+ authentication fails due to an error with the server, local authentication is used instead. If local authentication fails, no authentication is used, and the device automatically permits access.

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

The first method parameter (**tacacs+** in the previous example) 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 **Authentication Method Values** table.

TABLE 4 Authentication Method Values

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 Configuring Local User Accounts on page 20 .
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.

Specifying Different Servers for Individual AAA Functions

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

By default, the TACACS+ server performs all AAA functions. After authentication takes place, the server that performed authentication is used for authorization and accounting. If the authenticating server cannot perform the requested function, 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 until every server in the configured list has been tried.

To specify different TACACS+ servers for authentication, authorization, and accounting, use the **authentication-only**, **authorization-only**, or **accounting-only** parameters as shown in the following example. The example also specifies a different TACACS+ key for each server as well as a port to be used for authentication (AAA operations).

```
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
```

For TACACS+, the **auth-port** parameter specifies the TCP port number for the authentication port on the server. The default port number is 49.

For more information on the **tacacs-server host** command, refer to the *RUCKUS FastIron Command Reference*.

Setting Optional TACACS+ Parameters

You can set the following optional parameters in a 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 resends an authentication request when the TACACS+ server does not respond. The retransmit value can be from 1 through 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 determining the server is dead and trying to authenticate using the next server. The dead-time value can be from 1 through 5 seconds. The default is 3 seconds.
- **Timeout**: This parameter specifies how many seconds the RUCKUS device waits for a response from a TACACS+ server before either retrying the authentication request or determining that the TACACS+ servers are unavailable and moving on to the next authentication method in the authentication-method list. The timeout can be from 1 through 15 seconds. The default is 3 seconds.

- **Enable port-based VLAN access**: This option allows TACACS+ server management access only to clients connected to ports that are members of a specified port-based VLAN.

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 through 32 characters in length and cannot include any space characters.

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

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

When you display the configuration of the RUCKUS device, the TACACS+ keys are encrypted.

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

Setting the Retransmission Limit

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

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

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

Setting the Timeout Parameter

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

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

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 the user is authenticated.
- Command authorization consults a TACACS+ server to obtain authorization for commands entered by the user.

Configuring Exec Authorization

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

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

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

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

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

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. Consider the following TACACS+ server configuration.

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

In the previous 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. Refer to 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 and uses this A-V pair to determine the user privilege level.

Consider the following TACACS+ server configuration.

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

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

TABLE 5 RUCKUS Equivalents for Non-"foundry-privlvl" A-V Pair Values

Value for Non-"foundry-privlvl" A-V Pair	RUCKUS Privilege Level
15	0 (super-user)
From 14 - 1	4 (port-config)
Any other number or 0	5 (read-only)

In the previous example, 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.

Consider the following TACACS+ server configuration.

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

In the previous 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 obtain authorization for commands entered by the user.

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

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

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

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

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

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

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

If configured, command accounting is performed for these commands.

AAA Support for Console Commands

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

- Login prompt that uses AAA authentication, using authentication-method lists
- Exec authorization
- Exec accounting
- Command authorization
- Command accounting
- System accounting

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

The following example sends an Accounting Start packet to the TACACS+ accounting server when an authenticated user establishes a Telnet or SSH session on the RUCKUS device and an Accounting Stop packet when the user logs out.

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

Configuring TACACS+ Accounting for CLI Commands

You can configure TACACS+ accounting for CLI commands by specifying a privilege level for which 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, authorization is performed before accounting takes place. If authorization fails for the command, no accounting takes place.

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

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

Configuring TACACS+ Accounting for System Events

You can configure TACACS+ accounting to record system events on the RUCKUS device. System events include rebooting and changes to the active configuration.

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

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

Configuring an Interface as the Source for All TACACS+ Packets

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

Configuring TACACS+ for Devices in a 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+.

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

```
aaa authentication login default tacacs+ <- For Telnet/SSH login, use TACACS+ authentication
aaa accounting commands 0 default start-stop  tacacs+ <- Do TACACS+ accounting
                                                        for all device commands
aaa accounting exec default start-stop  tacacs+ <- Do TACACS+ accounting for Telnet/SSH
                                                  on connection and log out
aaa accounting system default start-stop  tacacs+ <- Use TACACS+ accounting to record
                                                     system events
ip address 10.10.6.56/255
tacacs-server host 255.253.255 <- Designates TACACS+ host
tacacs-server key 2 $d3NpZ0BVXFpJ  <- Defines TACACS+ key
```

TACACS+ Configuration Example

The following example configures a TACACS+ server and related operations.

```
device# configure terminal
/* First three lines define three authentication servers, port, function, and key */
device(config)# tacacs-server host 10.2.3.4 auth-port 49 authentication-only default key abc
device(config)# tacacs-server host 10.2.3.5 auth-port 49 authorization-only default key def
device(config)# tacacs-server host 10.2.3.6 auth-port 49 accounting-only default key ghi

/* Authenticate Telnet access using TACACS+ first. If that fails,
use local username and password. If that fails, permit access with no authentication. */
device(config)# aaa authentication login default tacacs+ local none
/* On successful login, enter CLI at the priveleged-EXEC level. */
device(config)# aaa authorization commands 0 default tacacs+
/* Use TACACS+ to check the user's privilege level for commands entered. */
device(config)# aaa authorization exec default tacacs+
/* Do TACACS+ accounting for SSH/Telnet, for all commands, and for all system events. */
device(config)# aaa accounting exec default start-stop tacacs+

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

Displaying 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+ information displayed by the **show aaa** command.

TABLE 6 Output of the show aaa Command for 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+ 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.

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

RADIUS Authentication

- RADIUS Security 37
- RADIUS Configuration Considerations 40
- Configuring RADIUS (Overview) 40
- Configuring Company-Specific Attributes on the RADIUS Server 41
- Identifying the RADIUS Server to the RUCKUS Device 42
- Configuring an SSL Profile for Use with RADIUS Server Hosts 43
- Specifying Different Servers for Individual AAA Functions 44
- RADIUS Security Configuration Example 45
- Mapping RADIUS Servers to Ports 46
- RADIUS Configuration Example 47
- Setting Up RADIUS over IPv6 47
- Setting RADIUS Parameters 48
- Configuring Detection of Dead RADIUS Servers 48
- Source Address Configuration for RADIUS Packets 49
- Configuring Authentication-method Lists for RADIUS 50
- RADIUS Authorization 51
- RADIUS Accounting 54
- Displaying RADIUS Configuration Information 56

RADIUS Security

You can use a Remote Authentication Dial In User Service (RADIUS) server to secure the following types of access to the RUCKUS switch:

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

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

FastIron supports TLS encryption for RADIUS server and client authentication.

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

RADIUS Authentication

When RADIUS authentication takes place, the following events occur.

1. A user attempts to gain access to the RUCKUS device by doing one of the following:
   - Logging into the device using Telnet, SSH, or the Web Management Interface
   - Entering the Privileged EXEC level or global configuration level of the CLI
2. The user is prompted for a username and password.
3. The user enters a username and password.

4. The RUCKUS device sends a RADIUS Access-Request packet containing the username and password to the RADIUS server.
5. The RADIUS server validates the RUCKUS device using a shared secret (the RADIUS key).
6. The RADIUS server looks up the username in its database.
7. If the username is found in the database, the RADIUS server validates the password.
8. If the password is valid, the RADIUS server sends an Access-Accept packet to the RUCKUS device, authenticating the user. Within the Access-Accept packet are three RUCKUS vendor-specific attributes that indicate the following:
   - The privilege level of the user
   - A list of commands
   - Whether the user is allowed or denied usage of the commands in the list

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

RADIUS Authorization

When RADIUS authorization takes place, the following events occur.

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

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

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

RADIUS Accounting

RADIUS accounting works as follows.

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

6. When the event is concluded, the RUCKUS device sends an Accounting Stop packet to the RADIUS accounting server.
7. The RADIUS accounting server acknowledges the Accounting Stop packet.

AAA Operations for RADIUS

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

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

AAA Security for Commands Pasted into the running-config

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

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

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

RADIUS Configuration Considerations

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

- For the RADIUS feature to function, 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.
- As an option, you can configure a RADIUS server as a port server, indicating that the server will be used only to authenticate users on ports to which it is mapped, as opposed to globally authenticating users on all ports of the device. Refer to Mapping RADIUS Servers to Ports on page 46.
- You can map up to eight RADIUS port servers to each port on the RUCKUS device. The port will authenticate users using only the RADIUS servers to which it is mapped. If there are no RADIUS servers mapped to a port, it will use the "global" servers for authentication. Refer to Mapping RADIUS Servers to Ports on page 46.
- You can select only one primary authentication method for each type of access to a device (CLI through Telnet, CLI Privileged EXEC and CONFIG levels). For example, you can select RADIUS as the primary authentication method for Telnet CLI access, but you cannot also select TACACS+ authentication as the primary method for the same type of access. However, you can configure different backup authentication methods for each access type.
- If your RUCKUS device is configured with multiple IP addresses, the lowest numbered IP address sources the device's RADIUS traffic by default. You may select a specific interface by configuring the **management source-interface protocol radius** command. If the selected interface has multiple IPs, the lowest numbered IP address is used.

Configuring RADIUS (Overview)

Complete the following steps to configure a RUCKUS device for RADIUS.

1. Configure RUCKUS vendor-specific attributes on the RADIUS server. Refer to Configuring Company-Specific Attributes on the RADIUS Server on page 41.
2. Identify the RADIUS server to the RUCKUS device. Refer to Identifying the RADIUS Server to the RUCKUS Device on page 42.
3. (Optional) Specify different servers for individual AAA functions. Refer to Specifying Different Servers for Individual AAA Functions on page 44.
4. (Optional) Configure the RADIUS server as a "port only" server. Refer to Mapping RADIUS Servers to Ports on page 46.
5. (Optional) Bind the RADIUS servers to ports on the RUCKUS device. Refer to Mapping RADIUS Servers to Ports on page 46.
6. (Optional) Adjust RADIUS parameters. Refer to Setting RADIUS Parameters on page 48.
7. (Optional) Set up dead server detection. Refer to Configuring Detection of Dead RADIUS Servers on page 48.
8. Configure authentication-method lists. Refer to Configuring Authentication-method Lists for RADIUS on page 50.
9. (Optional) Configure RADIUS authorization. Refer to RADIUS Authorization on page 51.
10. (Optional) Configure RADIUS accounting. Refer to RADIUS Accounting on page 54.

Configuring Company-Specific Attributes on the RADIUS Server

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

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

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

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

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

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

TABLE 7 RUCKUS Vendor-Specific Attributes for RADIUS

Attribute Name	Attribute ID	Data Type	Description
foundry-privilege-level	1	integer	Specifies the privilege level for the user. This attribute can be set to one of the following: **0** - **Super User level** - Allows complete read-and-write access to the system. This is generally for system administrators and is the only management privilege level that allows you to configure passwords. **4** - **Port Configuration level** - Allows read-and-write access for specific ports but not for global (system-wide) parameters. **5** - **Read Only level** - Allows access to the Privileged EXEC mode and User EXEC mode of the CLI but only with read access.
foundry-command-string	2	string	Specifies a list of CLI commands that are permitted or denied to the user when RADIUS authorization is configured. The commands are delimited by semi-colons (;). You can specify an asterisk (*) as a wildcard at the end of a command string. For example, the following command list specifies all **show** and **debug ip** commands, as well as the **write terminal** command: show *; debug ip *; write term*
foundry-command-exception-flag	3	integer	Specifies whether the commands indicated by the foundry-command-string attribute are permitted or denied to the user. This attribute can be set to one of the following: **0** - Permit execution of the commands indicated by foundry-command-string, deny all other commands. **1** - Deny execution of the commands indicated by foundry-command-string, permit all other commands.

TABLE 7 RUCKUS Vendor-Specific Attributes for RADIUS (continued)

Attribute Name	Attribute ID	Data Type	Description
foundry-access-list	5	string	Specifies the access control list to be used for RADIUS authorization. Enter the access control list in the following format. type=string, value="ipacl.[e|s].[in|out] = [acl-name | acl-number] separator macfilter.in = [acl-name | acl-number] Where: separator can be a space, newline, semicolon, comma, or null characater ipacl.e is an extended ACL; ipacl.s is a standard ACL.
foundry-MAC-authent-needs-802x	6	integer	Specifies whether or not 802.1x authentication is required and enabled. **0** - Disabled **1** - Enabled
foundry-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
foundry-coa-command	10	string	Specifies to perform CoA command dynamically on the port or host after the device or user is authenticated. **disable-port** - Disables the specified port. **reauth-host** - Re-authenticate the host specified by MAC address. **flip-port** - Brings the port up and down with some delay between the toggle. **modify-acl** - Replace the specified ACL with the session's existing ACL. Modify-ACL is supported with the Filter-Id (11) attribute. The IP ACL specified through the Filter-Id attribute replaces the session's existing ACL configuration.

Identifying the RADIUS Server to the RUCKUS Device

To use a RADIUS server to authenticate access to a RUCKUS device, you must identify the server to the RUCKUS device using the **radius-server host** command.

NOTE
The **radius-server host** and **key** parameters must be entered on the same command line to configure the ICX device to authenticate end devices through 802.1x or MAC authentication.

The **key** parameter 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 through 64 characters in length and cannot include any space characters.

The following example configures a RADIUS server with IP address 10.157.22.99 and RADIUS key ruckus as a Flexible authentication (dot1x macauth) server.

```
device# configure terminal
device(config)# radius-server host 10.157.22.99 acct-port 1813 default key ruckus dot1x mac-auth
```

Refer to the *RUCKUS FastIron Command Reference* for more information on the **radius-server** commands.

Configuring an SSL Profile for Use with RADIUS Server Hosts

You must configure an SSL profile, to be applied to the RADIUS server, for use in establishing a secure TLS connection. The SSL profile specifies the root (CA) certificate trustpoint and the remote domain name to be used in certification.

1. Name the SSL profile and enter profile configuration mode.

   ```
   device# configure terminal
   device(config)# ip ssl profile tls01
   ```

2. Specify the trustpoint (CA server) that will be associated with the profile.

   ```
   device(config-ssl-tls01)# trustpoint TLS-ABCD
   ```

3. Configure the remote domain name that the FQDN of the remote network peer certificate issues to the server. This is the 'reference identifier' that must appear in the network peer's certificate.

   NOTE
   The ICX device expects the 'reference identifier' value to be either in CN, or, if SAN is present, this value must be shown as a DNS name in the SAN.

   NOTE
   The remote domain name must match the CN or SAN. ICX devices do not support wild card bits in SAN extensions.

   ```
   device(config-ssl-tls01)# remotedomain ruckus.com
   device(config-ssl-tls01)# exit
   device(config)#
   ```

4. (Optional) Use the **show ip ssl profile** command to check the user SSL profile and device SSL profile information.

The following example configures the SSL profile tls01 and associates it with the trustpoint TLS-ABCD with ruckus.com as the remote domain name that the end user certificate issues to the server.

```
device# configure terminal
device(config)# ip ssl profile tls01
device(config-ssl-tls01)# trustpoint TLS-ABCD
device(config-ssl-tls01)# remotedomain ruckus.com
```

Specifying Different Servers for Individual AAA Functions

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

The following example specifies different RADIUS servers for authentication and accounting.

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

TLS and RADIUS

TLS-encrypted sessions support 802.1X authentication, MAC authentication, and Web authentication features in IPv4 and IPv6 networks. A previously configured SSL profile must be used for TLS-encrypted sessions for 802.1X authentication, MAC authentication, and Web authentication.

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

NOTE
As compared to TLS connections on logging host or TACACS+ servers, TLS connections on RADIUS servers require an extra TLS connection for accounting services.

You can use the **show ip ssl profile** command to verify the SSL profile that has been applied to the device.

```
device(config)# show ip ssl profile
SSL Profile Information:
**************************

Trustpoint Name    : DEVICE_TRUSTPNT
Remote Domain      : BTC3243M00J.switch-id.ruckuswireless.com
Signature Algorithm: sha256WithRSAEncryption

 Not Before: 2016 Feb 18 19:28:17
 Not After : 2036 Feb 18 19:28:17


Common Name        : RuckusRootCA
Organization       : Ruckus Wireless
Locality           : Sunnyvale
State              : California
Country            : US

Common Name        : RuckusRootCA
Organization       : Ruckus Wireless
Locality           : Sunnyvale
State              : California
Country            : US

-----BEGIN CERTIFICATE-----
MIIFszCCA5ugAwIBAgIIXNVhcWHHz5EwDQYJKoZIhvcNAQELBQAwZzEVMBMGA1UE
AwwMUnVja3VzUm9vdENBMRgwFgYDVQQKDA9SdWNrdXMgV2lyZWxlc3MxEjAQBgNV
BAcMCVN1bm55dmFsZTETMBEGA1UECAwKQ2FsaWZvcm5pYTELMAkGA1UEBhMCVVMw
HhcNMTYwMjE4MTkyODE3WhcNMzYwMjE4MTkyODE3WjBnMRUwEwYDVQQDDAxSdWNr
dXNSb290Q0ExGDAWBgNVBAoMD1J1Y2t1cyBXaXJlbGVzczESMBAGA1UEBwwJU3Vu
bnl2YWxlMRMwEQYDVQQIDApDYWxpZm9ybmlhMQswCQYDVQQGEwJVUzCCAiIwDQYJ
KoZIhvcNAQEBBQADggIPADCCAgoCggIBAMz6iReNwJmMZiobq2Cr39RW90vmso2f
ZEZ6HLFab8ih71L6+mzXoR8OMoQdVtmX5tRW5e/Nk0+6MHk2y1gDw/yIHGc3Ivc0
IHijTx0GFfP5zoKBcpUf9ccNNJnR+rgSWAFCWpDLJotRBf7Gg9smpKJ/Nvdkn8gE
5qnEOyCGwG8MEy6gjjAOckG0IC3h53pI7VekhKAxC4xrwUQD4EqLgVBisL8nCcAl
89ec8hZ78yGAxD/zNN51+UC+nzqP3jSNKe8Yzfv4teBpit/V5ueilw2x/R4Ys9GR
bdiYf8LtfVqDFuj/sLayTJXZNDZKXgnnvZBYP8WEb3fWWkcEknJi8CrtE1uYK1S1
Uo1OocYpJbwFOSZaBZTGal9744N6zXFEFChyYsVCW5Hn4xPfoZrFWWhx+nqhpsc3
HLte6yKG9u+YQWZdwItwaolj+6q6hTe3oleDyfxxgswh8oA7jERCBw8Gk01zL5yS
```

```
alx6ctnv417PTw9ZiDIfZqpYOu1rj8MoM9MMpqYjCSbMQthLJqvXfqC2iItt8TyP
I7XAeeR5+M3BqyvInMCz2cdZ464QXt2DyTS+WBB5SYsXD2sMAli7z9QhkIdCUVDR
6pW4G1ewAEFWcG+kzjJRWXDdCXmI4aGs769xvFh5S76+U5t6lC4O1yhZ7TJT9jvx
oQIJ9rjwPpV9AgMBAAGjYzBhMB0GA1UdDgQWBBQV2AsIvxliqdlVa33cp1HpD1Ti
eTAPBgNVHRMBAf8EBTADAQH/MB8GA1UdIwQYMBaAFBXYCwi/GWKp2VVrfdynUekP
VOJ5MA4GA1UdDwEB/wQEAwIBhjANBgkqhkiG9w0BAQsFAAOCAgEADdn/yh/+wNam
RAUG3v17IwC1uoAE5rSO9lc2k5cYIb+iT+HisY1QHRpqiZOhffRv2MZHR+YDuhL8
c/TcWbg4ElmjhgOZcE37kQysF8iHTGRmawGtGGPvjwLsP57t8Wnz3qAvRi2hdc/V
JwJB6FRBu9ESKdt1deQoZ0ccjE71wNWL1dwuovarFMPRLS/u7iLeTMGfIE200Pna
Ztjq2bsf33fVjL1pj10R1lLgStTDUSvL+IdjAaS2LO5PZowtPfTHaRgT7SDmmnWr
EC4SyNmP8PmAkBkHYWkVmgWj/VyDfnHcFEwa8dd5iQVMm+J+2J5oCkg3NZszjAoF
5RzOF+Vcp5FGiPUkNgCt7zg2MbvNLQIDJ8aAIS8GYKjnbZ4pezTw5K/y5f+qK5s/
ZmdOG8zMiCkP70lek3LJK7RSPfHEOEP2Ff89+LflDpW4U4gnUs+qIDaRiBk6NBm8
3FUk3np9agW3QhFgKLhctB2T5qWSfS83pRQFdiL0KcxllDAAHQULwl0i/3lPYwVJ
VvsH5BGYjKLz3MvyGxVUFh4SXKA4GEiELdwSb9PJXhHCQrJ1pHVq+D+PVdGwz2m8
feoGCwF3w3nJVrDGwLOzgm2f01aDHbJMlM1YcgW/ET6VbwjKus5eNlJh7YacbIqw
I7InlLanM0XvBvl6hDoEsV/zq7g/Cjo=
-----END CERTIFICATE-----
```

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, 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 until every server in the configured list has been tried.

The following example shows the default profile available in TPM devices (not user configured).

```
radius-server host 10.177.131.182 ssl-auth-port 2083 profile DEVICE_PROFILE default key 123qwe dot1x mac-
auth web-auth
```

The following example applies a previously configured SSL profile with a TLS-encrypted session for the RADIUS server.

```
device(config)# radius-server host 10.177.131.182 ssl-auth-port 2083 profile tls-profile default key radsec
dot1x mac-auth web-auth
```

In the user-configured example, the **ssl-auth-port** keyword specifies that the server is a RADIUS server running over a TLS-encrypted TCP session. The specified port, 2083, is the default destination TCP port number for RADIUS over TLS. The source port is arbitrary and is not specified. In the example, "tls-profile" is the name of the SSL profile. The keyword **default** indicates that the server can be used for both authentication and accounting operations. The authentication methods configured are represented by the keywords **dot1x**, **mac-auth**, and **web-auth**.

Only one auth-port or ssl-auth-port can be specified. If neither is specified, the default auth-port of 1812 is used for authentication, and 1813 is used for accounting with no TLS encryption.

RADIUS Security Configuration Example

The following example configures a RADIUS security trustpoint and associated RSA key pair.

```
device# configure terminal
device(config)# pki import key rsa tls-key pem url flash: test-client.key.pem  <--Key file must be imported
                                                                                  before configuring
                                                                                  trustpoint
device(config)# pki trustpoint tls
device(config-pki-trustpoint-tls)# rsakeypair key-label tls-key
device(config-pki-trustpoint-tls)# fingerprint CD:84:56:28:47:63:74:AF:80:80:3A:B1:16:5C:67:C7:04:9B:CB:C9
device(config-pki-trustpoint-tls)# exit
device(config)# pki import tls pem url flash: test-client.cert.pem
device(config)# pki import tls pem url flash: test.rootca.pem
device(config)# pki import key rsa tls-key pem url flash: test-client.key.pem
```

The following example configures the trustupoint and the remote domain for an SSL profile.

```
device(config)# ip ssl profile tls_profile
device(config-ssl-tls01)# trustpoint tls
device(config-ssl-tls01)# remotedomain icx_poc_14.commscope.com
device(config-ssl-tls01)# exit
```

For more information on SSL profiles, refer to Configuring an SSL Profile for Use with RADIUS Server Hosts on page 43.

The following example configures the RADIUS server to use the previously configured TLS profile.

```
device(config)# radius-server host 10.177.171.221 ssl-auth-port 2083 profile tls_profile default key radsec
dot1x mac-auth web-auth
```

Mapping RADIUS Servers to Ports

As an option, you can configure a RADIUS server to authenticate only ports mapped to it. A RADIUS server that is not explicitly configured as a RADIUS server per port is a global server that can be used to authenticate users on ports to which no RADIUS servers are mapped.

When you map one or more RADIUS servers to a port on a RUCKUS device, the port authenticates users using only the RADIUS servers to which the port is mapped.

When more than one server is mapped to a port, the port tries the servers in the order they are mapped until a server that can perform the requested function is found or until every mapped server in the list has been tried.

If there are no RADIUS servers mapped to a port, the port uses the global servers for authentication.

Keep the following points in mind when mapping ports to RADIUS servers.

- This feature works with 802.1X and MAC authentication only.
- You can define a maximum of eight RADIUS servers per RUCKUS device.
- You can map a RADIUS server to a physical port only. You cannot map a RADIUS server to a VE.

Complete the following steps to map a RADIUS server or a set of RADIUS servers to specific ports.

1. Enter global configuration mode.

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

2. (Optional) Configure one or more RADIUS servers to authenticate only ports that are explicitly mapped to it. Use the **port-only** keyword as part of each **radius-server host** command.

   ```
   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
   ```

   RADIUS servers 10.10.10.103 and 10.10.10.104 are configured with the **port-only** option so that they authenticate users only on ports to which the servers are mapped.

3. Configure 802.1x or MAC Auth as the authentication method to be used on a specific port.

   NOTE
   Refer to Configuring Flexible Authentication on page 200 for more information on configuring authentication on a port.

4. Enter interface configuration sub-mode for the port.

   ```
   device(config)# interface ethernet 3/1/1
   ```

5. Enter the **use-radius-server** command followed by the IP address of the authentication server for each RADIUS server to be mapped to the port.

   ```
   device(config-if-e1000-3/1/1)# use-radius-server 10.10.10.103
   device(config-if-e1000-3/1/1)# use-radius-server 10.10.10.110
   ```

   The example maps two RADIUS authentication servers to port 3/1/1.

The following examples map RADIUS servers to specific ports.

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

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

The following example enables MAC authentication on port 2/1/1 and configures authentication through the same RADIUS servers as in the previous example.

```
device# configure terminal
device(config)# authentication
device(config-authen)# mac-auth enable
device(config-authen)# mac-auth enable ethernet 2/1/1
device(config-authen)# exit
device(config)# interface ethernet 2/1/1
device(config-if-e1000-2/1/1)# use-radius-server 10.10.10.103
device(config-if-e1000-2/1/1)# use-radius-server 10.10.10.110
device(config-if-e1000-2/1/1)# end
```

RADIUS Configuration Example

The following example configures four RADIUS hosts.

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

The configuration has the following effect:

- RADIUS servers 10.10.10.103 and 10.10.10.104 are configured with the **port-only** option so that they authenticate users only on ports to which the servers are mapped. To map a RADIUS server to a port, refer to Mapping RADIUS Servers to Ports on page 46.
- RADIUS servers 10.10.10.105 and 10.10.10.106 are used to authenticate users on ports to which no RADIUS servers are mapped. For example, port e 9, to which no RADIUS servers are mapped, sends a RADIUS request to the first configured RADIUS server, 10.10.10.105. If the request fails, it goes to the second configured RADIUS server, 10.10.10.106. The request never goes to server 10.10.10.103 or 10.10.10.104 because these servers are configured as port servers.

Setting Up RADIUS over IPv6

RUCKUS devices can send RADIUS packets over an IPv6 network.

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

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

NOTE
The **radius-server host** command and the **radius-server key** command must be entered on the same command line to configure the ICX device to authenticate end devices through 802.1x or MAC authentication.

```
device# configure terminal
device(config)# radius-server host ipv6 2001:DB8::300 acct-port 1813 default key ruckus dot1x mac-auth
```

The example configures the RADIUS server with the IPv6 address 2001:DB8::300 and the RADIUS key ruckus for 802.1x and MAC authentication.

Setting RADIUS Parameters

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 **retransmit** parameter specifies the maximum number of retransmission attempts. When an authentication request times out, the RUCKUS device retransmits the request up to the maximum number of times configured.

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

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

   NOTE
   The **radius-server timeout** command is not applicable to RADIUS TLS connections.

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

   The example sets the RADIUS server to time out after 5 seconds.

2. To set the RADIUS retransmit limit, enter the **radius-server retransmit** command followed by a value from 1 through 5 as shown in the following example. The default is 2.

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

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

Configuring Detection of Dead RADIUS Servers

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

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

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

Complete the following steps to configure dead RADIUS server detection.

1. Enter global configuration mode.

```
device# configure terminal
```

2. (Optional) Adjust the time limit for server responses and maximum retries before the RADIUS server is declared dead.

NOTE
The **radius-server timeout** command is not applicable to RADIUS TLS connections.

NOTE
For more information, refer to Setting RADIUS Parameters on page 48.

```
device(config)# radius-server timeout 4
device(config)# radius-server retransmit 5
```

3. Enter the **radius-server test** command followed by the name of a non-existent user.

```
device(config)# radius-server test batman
```

4. (Optional) Adjust the test message interval. Enter the **radius-server dead-time** command followed by the desired value, from 1 through 5 minutes. The default dead-time for RADIUS is 2 minutes.

```
device(config)# radius-server dead-time 2
```

The following example configures dead server detection.

```
device# configure terminal
device(config)# radius-server test batman
device(config)# radius-server dead-time  3
device(config)# exit
device# show radius server
-------------------------------------------------------------------------------
Server                      Type      Opens     Closes   Timeouts    Status
-------------------------------------------------------------------------------
10.20.226.113                any        471        247          1   active
10.20.226.114                any        471        247          1   dead
```

Source Address Configuration for RADIUS Packets

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

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

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

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

The following example configures a virtual interface as the soucr IP address for the RADIUS client.

```
device(config)#  management source-interface ve 100 protocol radius
```

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

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

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

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

Refer to the *RUCKUS FastIron Command Reference* for details on the **management source-interface protocol radius** command.

Configuring Authentication-method Lists for RADIUS

You can use RADIUS to authenticate Telnet or SSH access and access to the Privileged EXEC and global configuration levels of the CLI. When you configure authentication-method lists for RADIUS, you must create an authentication-method list for Telnet or SSH CLI access and a second separate authentication-method list for access to the Privileged EXEC and global configuration levels of the CLI.

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

NOTE
For examples of how to define authentication-method lists for types of authentication other than RADIUS, refer to Authentication-Method List Overview on page 59.

Perform the following steps to configure authentication-method lists for remote and CLI access.

NOTE
Telnet and SSH are enabled by default in FastIron release 09.0.00 and later releases.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **aaa authentication login** command followed by the desired authentication-method list parameters.

   NOTE
   Refer to Authentication-Method Values on page 51 for available RADIUS authentication-method list parameters.

   The following example creates an authentication-method list for securing remote Telnet or SSH access to the CLI.

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

   In the example, the **default** parameter specifies that the methods listed on the same line form the default authentication-method list. The example configures RADIUS as the primary authentication method for securing remote Telnet or SSH access to the CLI. If RADIUS authentication fails due to a server error, local authentication is used instead. If local authentication fails, access is not granted.

3. Configure authentication methods for access to the Privileged EXEC and global configuration levels of the CLI. Enter the **aaa authentication enable** command followed by desired parameters.

   The following example configures authentication methods for securing access to the Privileged EXEC level and CONFIG levels of the CLI.

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

   The example configures RADIUS as the primary authentication method for securing access to the Privileged EXEC and CONFIG levels of the CLI. If RADIUS authentication fails due to a server error, local authentication is used instead. If local authentication fails, access to the Privileged EXEC and CONFIG levels of the CLI is not allowed.

Authentication-Method Values

The following table lists parameters available for configuring RADIUS authentication-method lists.

TABLE 8 Authentication-Method Values

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 Configuring Local User Accounts on page 20.
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.

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
```

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, the value in the foundry-privilege-level attribute is ignored, and the user is granted Super User access.

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

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 is allowed to issue a command he or she has entered.

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

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

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

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

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

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

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

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

Enabling RADIUS CoA and Disconnect Message Handling for Dynamic Authorization

Per RFC 5176, a Dynamic Authorization Client (DAC) can dynamically terminate or authorize RADIUS sessions authenticated through MAC, 802.1x, or Web authentication. The ability to manage sessions also allows previous policies or configurations to be replaced. When no DAC is configured, established RADIUS sessions end only if the user or device logs out.

You must enable Disconnect Messages (DMs) and Change of Authorization (CoA) message handling on the ICX device for use with a DAC. To enable RADIUS Disconnect Message and CoA handling, complete the following steps:

1. Enter global configuration mode.

```
device# configure terminal
```

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

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

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

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

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

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

RADIUS Disconnect Message and CoA Events

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

Supported IETF Attributes in RFC 5176

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

TABLE 9 Supported IETF Attributes

Attribute Name	Attribute Number	Description
NAS-IP-Address	4	IPv4 address of NAS
NAS-Identifier	32	The port where the session is terminated
NAS-IPv6-Address	95	IPv6 address of NAS
Calling-Station-Id	31	Link address from which sessions are connected
Filter-ID	11	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 cause attribute to provide more details on the possible cause of the problem. The format of this error cause attribute is the same as any other attribute, and the value field consists of a 4-byte integer.

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

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

TABLE 10 Error Cause 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

RADIUS Accounting

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

Configuring RADIUS Accounting for Telnet/SSH (Shell) Access

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

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

Configuring RADIUS Accounting for CLI Commands

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

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

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

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

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

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

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

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

Configuring RADIUS Accounting for System Events

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

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

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

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

RADIUS Accounting for 802.1X Authentication and MAC Authentication

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

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

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

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

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

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

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

Interim RADIUS Accounting Update for 802.1X Authentication and MAC Authentication

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

The interim update message is useful for billing longer sessions. Normally, a RADIUS server uses the information from Accounting Start and Accounting Stop packets to generate accounting information for billing or other purposes. Information such as session time and number of bytes transferred that are critical for billing are available only in the Accounting Stop packets. If the device becomes unavailable unexpectedly, the data required for billing the sessions initiated on the device is not recorded. In such scenarios, interim updates send useful information regarding a specific session to the RADIUS server at regular intervals. For more information about the RADIUS attributes that support interim updates, refer to the list of supported RADIUS attributes in Flexible Authentication.

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

Enabling RADIUS Accounting for 802.1X Authentication and MAC Authentication

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

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

```
device# configure terminal
```

2. Enable accounting for 802.1X authentication or MAC authentication.
   - Enable 802.1X authentication accounting.

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

   - Enable MAC authentication accounting.

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

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

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

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

```
                packets in=1 packets out=4
no connection
```

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

TABLE 11 Output of the show aaa Command for RADIUS

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

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

Authentication-Method Lists

- Authentication-Method List Overview..... 59

Authentication-Method List Overview

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 login password
- Local password for the Super User privilege level
- Local user accounts configured on the device
- Database on a TACACS+ server
- Database on a RADIUS server

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

NOTE
For information on configuring management access to the RUCKUS ICX switch based on IP addresses, refer to "Remote Access to the Switch" in the *RUCKUS FastIron Management Configuration Guide*.

In an authentication-method list for a particular access method, you can specify up to three authentication methods (local, TACACS+, and RADIUS). 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 TACACS+ or RADIUS server. Otherwise, the device will authenticate using only the locally based password for the Super User privilege level.
- If **aaa authentication login/snmp/web/enable default local** is enabled, the last local user cannot be deleted. The last local user can be deleted only when RADIUS or TACACS+ server is configured as the authentication method.

Examples of Authentication-method Lists

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

Example 1

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

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

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

Example 2

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

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

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

Example 2

To configure the device to use a RADIUS server first to authenticate Web access, enter the **aaa authentication web-server login default** command followed by the keyword radius (and, as an option, authentication methods you want to designate as alternatives).

> **NOTE**
> For Web access, only authentication is validated. Authorization and accounting are not validated.

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

The example configures the device to consult a RADIUS server first to authenticate Web access and then to consult the local user accounts if the RADIUS server is unavailable.

Authentication Methods Available

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

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

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

TABLE 12 Authentication method values

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 Configuring Local User Accounts on page 20.

TABLE 12 Authentication method values (continued)

Method Parameter	Description
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 37.

Default Authentication Values

By default, the following AAA authentication methods are configured:

- aaa authentication login default local
- aaa authentication snmp-server default local
- aaa authentication web-server default local

ICX Digital Certificates

- Overview of ICX Digital Certificates..... 63
- SSL Security..... 63
- TLS Support..... 65

Overview of ICX Digital Certificates

A RUCKUS ICX device requires a digital certificate to use certain applications. A digital certificate is required, for example, to use the ICX web interface or to monitor or configure an ICX device from SmartZone via ICX-Management. When a digital certificate is not present, these applications will not start or will not function as expected.

A valid certificate can be obtained in one of several ways.

- **TPM certificate** - All new RUCKUS ICX devices are shipped from the factory with a TPM certificate. The Trusted Platform Module (TPM) provides a secure computing environment implemented in secure hardware.
- **IP SSL certificate** - You can use available commands to import or remove an SSL certificate. Refer to SSL Security on page 63 for more information.
- **External certificate** - ICX devices can, when required, copy a certificate from an external source using available tftp commands.

Once a valid certificate is present, it remains available, unless the user erases startup configuration or uses a command to zeroize (clear) the certificate.

When no certificate is present, the RUCKUS ICX device is unable to use applications that require a certificate.

When more than one certificate is stored in the RUCKUS ICX device, the device selects the certificate for use based on the following order of priority:

1. User-defined (SSL) certificate
2. TPM certificate
3. non-TPM (auto-generated legacy) certificate

SSL Security

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

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

Configuring SSL consists of the following tasks:

1. If necessary, enabling the SSL server on the RUCKUS ICX device

   NOTE
   By default https is enabled, but if https is turned off and a certificate is generated, then https will not be turned on automatically.

2. If necessary, importing an RSA certificate and private key file from a client

Enabling the SSL Server on the Device

HTTPS is enabled by default. If necessary, you can disable it. If HTTPS has been disabled, you can re-enable it. HTTP, likewise, can be disabled and re-enabled.

The following commands disable HTTPS.

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

To re-enable the SSL server to use HTTPS on the RUCKUS ICX device, enter the following command.

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

Use the **http** parameter with the **web-management** or **no web-management** command to enable or disable HTTP servers.

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

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

Importing Digital Certificates and RSA Private Key Files

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

NOTE
RUCKUS ICX devices support a maximum of 10 intermediate CA's for PKI or SSL authentication, excluding the ROOT CA.

If you choose to import an RSA certificate and private key file, you can use TFTP or SCP to transfer the files.

The following example copies the client certificate "client_cert.pem" from the TFTP server at IP address 10.1.1.1 to the ICX file "ssl-client-cert".

```
ICX# copy tftp flash 10.1.1.1 client_cert.pem  ssl-client-cert
```

The following example copies the client private key file "client_cert.pem" from the TFTP server to the ICX file "ssl-client-private-key".

```
ICX# copy tftp flash 10.1.1.1 client_cert.pem  ssl-client-private-key
```

The following example copies the root, or trusted, certificate "root_cert.pem" from the TFTP server to the ICX target file "ssl-trust-cert".

```
ICX# copy tftp flash 10.1.1.1 root_cert.pem ssl-trust-cert
```

The following examples provide SCP alternative commands for copying the client certificate, private key, and root certificates.

The following example uses SCP to copy the client certificate from the remote SCP server at IP address 10.1.1.1.

```
ICX# copy scp flash 10.1.1.1 client_cert.pem  ssl-client-cert
```

The following example uses SCP to copy the client private key file from the remote server.

```
ICX# copy scp flash 10.1.1.1 client_cert.key.pem  ssl-client-private-key
```

The following example uses SCP to copy the root, or trusted, certificate from the remote server.

```
ICX# copy scp flash 10.1.1.1 root_cert.pem ssl-trust-cert
```

Displaying Information about ICX Digital Certificates

You can use one of the following commands to check for the presence of an SSL certificate on an ICX device and view its contents:

- **show ip ssl certificate** - displays the user certificate imported with the **copy tftp flash** command
- **show ip ssl device-certificate** - displays the TPM or non-TPM device certificate stored on the ICX device

The following example displays information for a device certificate.

```
device# show ip ssl device-certificate
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 238779085 (0xe3b7acd)
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=US, ST=fe044db7a0ec05cf9736bfbcc2e186a76da5a13e49b1f12c8717e5c5bf5c32f2, L=10.176.156.30,
O=cc:4e:24:8c:67:e8, OU=JLSAWZOIFZMD, CN=ICX
        Validity
            Not Before: Dec  3 22:40:24 2019 GMT
            Not After : Nov 19 22:40:24 2079 GMT
        Subject: C=US, ST=fe044db7a0ec05cf9736bfbcc2e186a76da5a13e49b1f12c8717e5c5bf5c32f2,
L=10.176.156.30, O=cc:4e:24:8c:67:e8, OU=JLSAWZOIFZMD, CN=ICX
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:9f:87:35:01:dd:c3:63:52:7b:9d:aa:13:b7:39:
                    a9:0a:12:51:84:6e:57:ed:62:65:b7:79:31:72:35:
                    08:9a:d8:36:8b:f3:c8:76:47:90:5f:88:37:bc:6b:
                    1d:1f:5c:fd:0e:94:2d:7b:3a:54:d0:17:3c:96:d7:
                    be:a5:d8:0a:9c:54:08:08:30:06:84:a3:cb:1c:9f:
                    e0:ab:25:ac:59:02:7e:7b:cd:c2:bf:58:8d:63:09:
```

TLS Support

TLS 1.2 is supported by default.

The TLS version cannot be changed through the CLI. The TLS version is decided based on the server support. For devices that act as an SSL server or HTTPS server, the default connection uses TLS1.2. The system will default to a lower version if necessary.

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

```
device# show ip ssl
Session  Protocol Source IP               Source Port  Remote IP               Remote Port
Profile
1        TLS1_2   10.177.131.18           32952        10.177.131.216          2083         tls_profile
```

Secure Shell (SSH)

- SSH Version 2 Overview....67
- Configuring SSHv2....69
- Optional Parameters Overview....74
- Terminating an Active SSH Connection....76
- SSHv2 Client....76
- Enabling SSH Access....78
- Displaying SSH Information....78

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. You can log into and configure the device using a publicly or commercially available SSH client program, just as you can with Telnet. However, unlike Telnet, which provides no security, SSH provides a secure, encrypted connection to the device.

The RUCKUS SSHv2 implementation is compatible with all versions of the SSHv2 protocol (2.1, 2.2, and so on). At the beginning of an SSH session, the RUCKUS ICX device negotiates the version of SSHv2 to be used. The highest version of SSHv2 supported by both the RUCKUS ICX device and the client is used for the session. Once the SSHv2 version is negotiated, the encryption algorithm with the highest security ranking is selected for the session.

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

NOTE
The SSH feature is integrated with the latest opensource OpenSSH.

SSHv2 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 SSHv2 Clients

The following SSH clients have been tested with SSHv2:

- SSH Secure Shell 3.2.3
- VanDyke SecureCRT 5.2.2
- F-Secure SSH Client 5.3 and 6.0
- PuTTY 0.62

NOTE
When you use PuTTY on a Windows system, an SSH session may drop if left idle for more than 45 minutes.

- OpenSSH 4.3p2
- RUCKUS ICX SSH Client

NOTE
Supported SSH client public key sizes are 256 bits and 384 bits for ECDSA keys and 2048 bits for RSA keys.

SSHv2 Supported Features

SSHv2 (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.

RUCKUS SSHv2 supports the following options:

- Key exchange methods for establising an SSH connection are **diffie-hellman-group14-sha1** and **diffie-hellman-group14-sha256**. By default, when an SSH client connects to an ICX device acting as an SSH server, first priority is given to the diffie-hellman-group14-sha256 key exchange method. To change active support, use the **ip ssh key-exchange-method** command as described in Setting Optional Parameters on page 75.
- The supported public key algorithms are ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, rsa-sha2-256, and rsa-sha2-512.
- 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 simultaneous inbound SSH connections are supported.
- Five simultaneous outbound SSH connections are supported.

SSHv2 Unsupported Features

The following are not supported with SSHv2:

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

SSHv2 Authentication Types

The RUCKUS implementation of SSHv2 supports the following types of user authentication:

- ECDSA and 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 RADIUS server.
- Interactive-authentication, where keyboard entry is used in response to challenge-response questions, which may include username and password.

Configuring SSHv2

You can configure the device to use any combination of the available SSHv2 Authentication Types on page 68. The SSH server and client negotiate which type to use.

To configure SSHv2, follow these steps:

1. Generate ECDSA or RSA host key, and private key pair for the device.

   Refer to Enabling and Disabling SSH by Generating and Deleting Host Keys on page 69.
2. Configure ECDSA or RSA challenge-response authentication.

   Refer to Configuring RSA Challenge-Response Authentication on page 71.
3. Set optional parameters.

   Refer to Setting Optional Parameters on page 75.

Enabling and Disabling SSH by Generating and Deleting Host Keys

To enable SSH, generate an ECDSA or RSA host key on the device. The SSH server on the RUCKUS ICX device uses this key, along with a dynamically generated server 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 cannot be started from clients until a host key is generated. After a host key is generated, clients can start sessions.

To disable SSH, 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 deleted, it is deleted from the flash memory of all management modules.

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

Generating an ECDSA or RSA Key Pair

A command to generate a host key pair places the key pair in flash memory and enables SSH on the ICX device, if it is not already enabled.

To generate host key pairs, complete the following steps:

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```
2. Generate RSA key pair.

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

   To generate a modulus 2048-bit key, enter the following command.

   ```
   device(config)# crypto key generate rsa modulus 2048
   ```
3. Generate ECDSA key pair.

   ```
   device(config)# crypto key generate ec size 384
   ```

Deleting ECDSA and RSA Key Pairs

You can delete both ECDSA and RSA key pairs with the **crypto key zeroize** command, or you can specify a key pair to be deleted. When a host key is deleted, it is deleted from the flash memory of all management modules. If all key pairs are removed from the flash memory, SSH will get disabled.

To delete both ECDSA and RSA key pairs from the flash memory, complete the following steps:

1. Enter global configuration mode.

```
device# configure terminal
```

2. Delete both ECDSA and RSA key pairs from the flash memory.

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

You can also specify a key pair if you want to delete only a particular key pair.

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

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

To delete the ECDSA host key pair from flash memory, enter the following command.

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

The following example shows the steps to generate ECDSA key pair, make an outbound connection from VM(Linux) to the device using ECDSA 384 bit, and delete the keys.

1. Generate an ECDSA key pair.

```
device(config)# crypto key generate ec size 384
```

2. After the key is generated successfully, make an inbound connection from VM (Linux) to the device with configured local user or remote user.
   a. Connect to the Management IP.

```
root@iaas-l-82:~# ssh -o HostKeyAlgorithms=ecdsa-sha2-nistp384 super@10.176.156.48
Password:
SSH@ICX7x50 Router>
```

   b. Connect to the Management IP.

```
root@iaas-l-82:~# ssh -o HostKeyAlgorithms=ecdsa-sha2-nistp384 super@10.176.156.48
Password:
~ #
```

3. Delete all host key pairs.

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

Providing the Public Key to Clients

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

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

Configuring RSA Challenge-Response Authentication

With RSA challenge-response authentication, a collection of client public keys are stored on the RUCKUS ICX device. Only clients that have a private key that corresponds to one of the stored public keys can gain access to the RUCKUS ICX device using SSH.

With RSA challenge-response authentication enabled, the following events occur when a client tries to access the RUCKUS ICX device using SSH:

1. The client sends its public key to the RUCKUS ICX device.
2. The RUCKUS ICX device compares the client public key to those stored in memory.
3. If there is a match, the RUCKUS ICX device uses the public key to encrypt a random sequence of bytes.
4. The RUCKUS ICX 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 ICX device.
7. The RUCKUS ICX 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 RSA challenge-response authentication consists of the following steps:

   - Importing Authorized Public Keys into the ICX Device on page 71
   - Enabling RSA Challenge-Response and Password Authentication on page 72

Importing Authorized Public Keys into the ICX Device

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

NOTE
The keys must be in RFC 4716 format before import.

1. Collect one public RSA key from each client to be granted access to the RUCKUS ICX device, and list the keys in one file.

   The following example shows a public key file containing one public key.

   NOTE
   The public key file may contain up to 16 keys.

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

```
---- BEGIN SSH2 PUBLIC KEY ----
Comment: "2048-bit RSA, converted from OpenSSH"
AAAAB3NzaC1yc2EAAAABIwAAAQEA0pt94yJmKwPfPZnxxYSSlaVaaqWgRM79EfRXf2XUrS
8e4hx8B1MmQEdye1oJrntvA8LyVUIepOdbc874i4259mtzSXx+cfZW0/QeJggT/1zE82+n
w7o6gGqNsE+XsTI2bi6KU4AlwIWULce74yfQY9/amy38ZZPCesKKurH4+2m/Ba693E9ilP
nJ0BIQidn+I8hARUGayrOTrx/e2YkdC+2aNh6mSl7KDiRyj8WBV3F5z5f5rYBL/WoJ2beo
R3L6H6wHXP8dZlF4IqeVxeIimkFTzMEE8r/wHCnhewetnDy3iJAgr0TXTicJlQpb1MCBkB
XaynjuDYSf4Kmgn8znaQ==
---- END SSH2 PUBLIC KEY ----
```

2. Enter the **copy tftp flash** command followed by the IP address of the TFTP server and the public key filename, and use the **ssh-pub-key-file** keyword, to import the public key file into the active configuration of the RUCKUS ICX device from a TFTP server.

   The following example copies the public key file pkeys.txt to the ICX device from the TFTP server at IP address 10.168.1.234.

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

3. (Optional) Enter the **show ip client-pub-key** command to confirm that the public key file has loaded correctly as shown in the following example.

   ```
   device# show ip client-pub-key
   ---- BEGIN SSH2 PUBLIC KEY ----
   Comment:  "2048-bit RSA, converted from OpenSSH"
   AAAAB3NaC1yc2EAAAABIwAAAQEA0pt94yJmKwPfPZnxxYSS1aVaaqWgRM79EfRXf2XUrs
   834hx881MmQedye1oJrntvA8LyVUIepOdbc874i4259mtSXx+cfZW0/QeJggT/1zE82+n
   w706gGqNsE+XsT12bi6KU4Al2IWULce74yfQY9/amy38ZPCesKKurH4+2m/Ba69391lp
   nJ0BIQidn+I8hARUGayrOTrx/e2^kdC+2aNh6mS17KDiRyj8WBV3F5z5f5rlYBL/WoJ2beo
   R3L6H6wHXP8dZ1F4IqeVxeIimkFTzMEE*r/wHCnhewetnDy3iJAgr0TXTicJ1Qpb1MCBkB
   XaynjuDYSf4Kmgn8znaQ==
   ---- END SSH2 PUBLIC KEY ----
   ```

Enabling RSA Challenge-Response and Password Authentication

After the SSH server on the device negotiates a session key and encryption method with the connecting client, user authentication takes place. On RUCKUS ICX devices, SSH supports the following authentication methods, used separately or together:

- **Public-key**: Enables the RSA key pair method, in which the public and private key are checked for a match before the client is authenticated.
- **Password**: Allows the user to log in with username and password. Users are prompted for a password when they attempt to log in. If there is no user account that matches the username and password supplied by the user, the user is not granted access.
- **Interactive**: On the RUCKUS ICX device, a form of challenge-response in which the username and password serve as the challenge and response.

  NOTE
  To disable username and password authentication, you must disable both the password and interactive methods of authentication.

  NOTE
  If you disable all forms of authentication, the SSH server is disabled.

All three authentication methods are enabled by default. Perform the following steps if necessary to change challenge-response configuration.

1. (Optional) To check the current challenge-response authentication settings, enter the **show ip ssh config** command.

```
device# show ip ssh config
SSH server                : Disabled
SSH port                  : tcp\22
Host Key                  :
Encryption                : aes256-cbc, aes192-cbc, aes128-cbc, aes256-ctr, aes
192-ctr, aes128-ctr, 3des-cbc
Permit empty password     : No
Authentication methods    : Password, Public-key, Interactive
Authentication retries    : 3
Login timeout (seconds)   : 120
Idle timeout (minutes)    : 0
Strict management VRF     : Disabled
SCP                       : Enabled
SSH IPv4 clients          : All
SSH IPv6 clients          : All
SSH Client Keys           : RSA(0)
Client Rekey              : 200 Minute, 0 KB
Server Rekey              : 250 Minute, 0 KB
```

   The example shows public-key, password, and interactive authentication are enabled.

2. If necessary, enable public-key authentication.

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

3. If desired, enable password authentication.

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

   When password authentication is enabled, a password is required unless the "allow empty password" option has been enabled.

4. If desired, enable interactive keyboard authentication.

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

   NOTE
   The **no** form of any of the previous commands disables that form of authentication.

The following example re-enables public-key authentication.

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

The following example disables username and password authentication but enables public key authentication.

NOTE
One authentication method must be enabled. If you attempt to disable all three authentication methods, the action is prevented, and an error message is displayed.

```
device# configure terminal
device(config)# ip ssh key-authentication yes
device(config)# ip ssh interactive-authentication no
device(config)# ip ssh password-authentication no
```

Deleting the Public Keys

A RUCKUS ICX device can store up to 16 keys. It is a good practice to remove old keys that are no longer in use.

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

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

   The example deletes the public key file "pkeys" from the RUCKUS ICX device.

2. To clear the client public key from the RUCKUS ICX device buffer, enter the following command.

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

Optional Parameters Overview

You can adjust the following SSH settings on the RUCKUS device.

- The user authentication method: For SSH connections, the RUCKUS ICX device can use several authentication types together or individually. Refer to SSHv2 Authentication Types on page 68.
- The number of SSH authentication retries: By default, the RUCKUS ICX device tries five times to negotiate a connection with the host. The number of authentication retries can be configured as 1 through 5. Open SSH counts the password, public key, and kbd inter-authentication attempts as authentication retries.
- Key exchange method: RUCKUS ICX SSHv2 offers diffie-hellman-group14-sha1 and diffie-hellman-group14-sha256 as the key exchange methods for establishing an SSH connection. By default, when SSH clients connect, the ICX device as SSH server gives diffie-hellman-group14-sha256 first priority. The diffie-hellman-group14-1 and diffie-hellman-group14-sha256 algorithms provide enhanced encryption of shared secrets between two devices and safeguard critical security parameters. To change active support, use the **ip ssh key-exchange-method** command as described in Setting Optional Parameters on page 75.
- The port number for SSH connections: By default, SSH traffic is carried on TCP port 22. You can change this port number; however, you must configure SSH clients to connect to the new port. You must also be careful not to assign SSH to a port that is used by another service.
- 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 by default for a response from the client. If there is no response from the client after 120 seconds, the SSH server disconnects. You can configure a timeout value from 1 through 120 seconds.
- Source interface for all SSH traffic from the device: By default, the management port is used as the source interface for all SSH traffic from the RUCKUS ICX device. You can instead designate an Ethernet port, virtual interface, or loopback interface.
- The maximum idle time for SSH sessions: By default, SSH sessions time out after 5 minutes of inactivity. You can set the amount of time an SSH session can be inactive before the RUCKUS device closes it. An idle time of 0 minutes means that SSH sessions never time out. The maximum idle time for SSH sessions is 240 minutes.
- SSH rekey: In an SSH2 implementation, if an SSH session is authenticated and established, it remains connected until the user closes it or until the configured idle time limit is reached. Prolonged use of the session key negotiated at connection poses several security issues and exposes the SSH connection to man-in-middle attacks. To safeguard the SSH connection from security vulnerability, new keys should be exchanged frequently for existing SSH sessions.

SSH Rekey Configuration Notes

SSH rekeying is the process of exchanging the session keys at a configured interval, either after a time limit or a data limit for an SSH session. Rekeying can be initiated by both the SSH client and SSH server. While the key exchange renegotiation is taking place, data is not passed through the SSH connection. The algorithm that was used at connection startup is used during rekeying. SSH rekey exchange is supported on OpenSSH-based client version 7.5 or later.

Keep the following items in mind when configuring SSH rekeying.

- SSH sessions established without the rekey configuration will not have the rekey functionality.
- When SSH rekeying is enabled, the existing SSH session will not have the rekey functionality until the rekey exchange occurs from the other side.
- Removing the rekey configuration disables the SSH rekey for existing SSH sessions.
- When the rekey configuration is changed, the behavior of the existing session will not be affected until the next rekey exchange for the corresponding session.
- When the next rekey exchange occurs, the related data and time parameters of the corresponding SSH session will reset to the configured rekey value.
- The encryption method must not be modified during SSH rekeying.

Setting Optional Parameters

Perform any of the following steps if necessary to change optional SSH parameters.

1. (Optional) Specify one or more key exchange methods. By default, diffie-helman-group14-sha256 is prioritized. The diffie-helman-group-14-sha1 option is also available.

   The following example adds and prioritizes diffie-helman-group14-sha256 if the option has been disabled.

   ```
   device(config)# ip ssh key-exchange-method dh-group14-sha256
   ```

   NOTE
   High CPU usage is expected while establishing SSH sessions with the diffie-hellman-group14-sha1 key exchange method.

   NOTE
   The diffie-helman-group14-sha1 option is always present and cannot be removed.

2. (Optional) Assign a new port to carry SSH traffic.

   The following example re-assigns port 2200 for SSH traffic.

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

   NOTE
   If you change the SSH port number, RUCKUS recommends that you change it to a port number greater than 1024.

3. (Optional) Specify an SSH connection timeout value from 1 through 120 seconds. The default is 120 seconds.

   The following example configures an SSH connection timeout of 60 seconds.

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

4. (Optional) Specify an interface type to be used for the SSH connection.

   The following example sets Ethernet port 1/2/4 as the SSH source interface on the RUCKUS ICX device.

   ```
   device(config)# ip ssh source interface ethernet 1/2/4
   ```

5. (Optional) Set the idle time for SSH sessions. The default is 2 minutes.

   NOTE
   The **cli timeout** command setting controls CLI, rconsole (remote console), Telnet, and SSH sessions.

   The following example configures SSH sessions to never time out due to inactivity.

   ```
   device(config)# cli timeout 0
   ```

   The following example configures SSH sessions to time out after 30 minutes of inactivity.

   ```
   device(config)# cli timeout 30
   ```

6. (Optional) Configure the interval for SSH rekey exchange.

   The following example sets the rekey exchange interval to 5 minutes.

   ```
   device(config)# ip ssh rekey time 5
   ```

Terminating an Active SSH Connection

NOTE
When you as administrator terminate an active SSH connection, you and any other user on that connection are logged off.

To terminate one of the active SSH connections, enter the **kill** command followed by the session number, as shown in the following example.

```
device# kill ssh 1
```

Refer to the *RUCKUS FastIron Command Reference* for more information on the **kill** command.

SSHv2 Client

The SSHv2 client allows you to connect from a RUCKUS ICX device to an SSHv2 server, including another RUCKUS ICX device that is configured as an SSHv2 server. You can start an outbound SSHv2 client session while you are connected to the device by any connection method (SSHv2, Telnet, console). RUCKUS ICX devices support one outbound SSHv2 client session at a time.

The following SSHv2 client features are supported:

- Encryption algorithms, in the order of preference:
  - aes256-ctr
  - aes192-ctr
  - aes128-ctr
  - aes256-cbc
  - aes192-cbc
  - aes128-cbc
  - 3des-cbc
- SSHv2 client session authentication algorithms:
  - Password authentication
  - Public key authentication
- Message Authentication Code (MAC) algorithm: hmac-sha1
- Key exchange algorithm: diffie-hellman-group14-sha1 or diffie-hellman-group14-sha256

- 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.

NOTE
The ICX SSH client does not support compression algorithms.

Enabling the SSHv2 Client

To use the SSHv2 client, you must first enable the SSHv2 server on the device. Refer to Configuring SSHv2 on page 69.

When the SSHv2 server is enabled, you can use the SSH client to connect to an SSH server using password authentication.

Configuring SSHv2 Client Public Key Authentication

To use an 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.

Generating and Deleting a Client RSA Key Pair

The following example generates a client RSA key pair with a modulus size of 2048 bits.

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

The following example deletes the RSA host key pair.

```
device# configure terminal
device(config)# no crypto key client zeroize rsa
```

NOTE
The **crypto key client zeroize** command without additional parameters removes all RSA keys from flash memory, thereby disabling SSH.

Exporting Client Public Keys

An RSA key is stored in the $$sshrsapub.key file in the flash memory of a RUCKUS ICX device.

To copy key files from flash to a TFTP server, you can use the **copy flash tftp** command.

You must copy the public key to the SSH server. If the SSH server is a RUCKUS ICX device, refer to Importing Authorized Public Keys into the ICX Device on page 71.

Establishing an SSHv2 Client Connection

To start an SSHv2 client connection to an SSHv2 server using password authentication, enter the **ssh** command followed by the IPv4 or IPv6 address of the host, as shown in the following example.

```
device# ssh 10.10.10.2
```

The example establishes a connection with the SSH host at IPv4 address 10.10.10.2.

To start an SSHv2 client connection to an SSHv2 server using public key authentication, enter the **ssh** command, the IP address of the host, the keyword **public-key** and the type of public key to be used, as shown in the following example.

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

The example establishes an SSH connection with the host at IPv4 address 10.10.10.2 using RSA public key authentication.

For more information on the **ssh** command, refer to the *RUCKUS FastIron Command Reference*.

Enabling SSH Access

By default, there are no restrictions on access for management protocols, including SSH. To allow SSHv2 access to a RUCKUS ICX device, you must generate a Crypto Key.

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

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **crypto key generate** command to generate the crypto key to enable SSH.

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

The following example generates a client RSA key pair with a modulus size of 2048 bits.

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

The following example generates an elliptical key pair named testkey with the default size of 384 bits.

```
device# configure terminal
device(config)# crypto key generate ec label testkey
```

Displaying SSH Information

To display information about SSH connections, enter the **show ip ssh** command.

```
device# show ip ssh

Connection  Version   Username  IP Address
Inbound:
1           SSH-2     Raymond   10.120.54.2
Outbound:
6           SSH-2     Steve     10.37.77.15
SSH-v2.0 enabled; hostkey:  RSA(2048)
```

To display more detailed information about SSH connections, enter the **show who** command.

```
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
```

To display SSH configuration information, enter the **show ip ssh config** command.

```
device# show ip ssh config

SSH server                 : Disabled
SSH port                   : tcp\22
Host Key                   :
Encryption                 : aes256-cbc, aes192-cbc, aes128-cbc, aes256-ctr, aes
192-ctr, aes128-ctr, 3des-cbc
Permit empty password      : No
Authentication methods     : Password, Public-key, Interactive
Authentication retries     : 3
Login timeout (seconds)    : 120
Idle timeout (minutes)     : 0
Strict management VRF      : Disabled
SCP                        : Enabled
SSH IPv4 clients           : All
SSH IPv6 clients           : All
SSH Client Keys            : RSA(0)
Client Rekey               : 200 Minute, 0 KB
Server Rekey               : 250 Minute, 0 KB
```

SCP Client Support

- SCP Client 81
- SCP Client Support Limitations 81
- Supported SCP Client Configurations 82
- Downloading an Image from an SCP Server 82
- Uploading an Image to an SCP Server 82
- Uploading Configuration Files to an SCP Server 83
- Downloading Configuration Files from an SCP Server 83
- Secure Copy with SSH2 83

SCP Client

Secure copy (SCP) supports file transfer between a local device and a remote host. It combines the file-transfer element of BSD remote copy (RCP) with the authentication and encryption provided by the Secure shell (SSH) protocol.

The SCP client feature on RUCKUS ICX devices helps to transfer files to and from the SCP server and maintains the confidentiality of the data being transferred by blocking packet sniffers from extracting valuable information from the data packets. You can use SCP client to do the following:

- 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

The 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 timestamps, as part of file data when you use SCP client to upload files. The SCP client supports the same copy features as the TFTP client feature on FastIron devices, but the SSH2 protocol secures data transfer.

SCP Client Support Limitations

SCP client sessions are limited by file size and by whether other SCP client sessions are running and by whether SC server sessions are in progress.

The following limitations apply to SCP client sessions:

- An SCP copy of the running or startup configuration file from a RUCKUS ICX device to Linux WS 4 or 5 may fail if the configuration size is less than 700 bytes.
- Only one SCP client session is supported at a time.
- An SCP client session cannot be initiated if an SCP server session is in progress.
- An SSH client outbound session cannot be initiated if an SCP client session is in progress from the same terminal.
- Uploading and downloading public or private key files is not supported.
- Downloading signature files is not supported.
- When transferring files between devices under test (DUTs), the following limitations apply:
  - When using a binary image copy to transfer files between DUTs, you should configure the **flash:primary** keyword rather than the **primary** keyword because the SCP server does not support remote-filename aliases. See the description of the **copy scp flash** or the **copy flash scp** command for more information.
  - Be sure to download the compatible configurations when you transfer startup configuration or running configuration files copy between DUTs because the overwrite option is restricted.
  - Copying power over Ethernet (POE) firmware between two DUTs is not supported.

- During image copy between two mixed stacking units, KX image copy is not supported and the KX image cannot be uploaded from the stack 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

The 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 }** command to generate or delete an SSH-client key pairs. The SSH-client public key is stored in the file - $$sshdsapub.key.
  - You can use the **crypto key client generate rsa [modulus 2048]** command to generate an SSH-client-RSA key pair. The SSH-client-RSA public key is stored in the file $$sshrsapub.key.
  - You can use the **crypto key client zeroize rsa** command to delete an SSH-client-RSA key pair.

Beginning with 8.0.30d release, the SCP file transfer speed over high latency connections is increased.

Downloading an Image from an SCP Server

Securely download image files from a secure copy (SCP) server.

Copy an image from the SCP server to a device as shown in the following example.

```
device# copy scp flash 10.20.1.1 SPR09000.bin primary
device# copy scp flash 10.20.1.1 SPR09000.bin secondary
```

The example copies a primary and secondary image from the SCP server with the IP address 10.20.1.1.

For more information on the **copy scp flash** command, refer to the *RUCKUS FastIron Command Reference*.

Uploading an Image to an SCP Server

To securely upload image files to a secure copy (SCP) server, copy an image from a device to the SCP server.

```
device# copy flash scp 10.20.1.1 SPR09000.bin primary
device# copy flash scp 10.20.1.1 SPR09000.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-78-startup
```

   You are prompted for a login ID and password. Then the startup configuration file is uploaded to the SCP server and you are notified when the transfer is complete.

```
 Login:tjones

 Password:
Parameter Validation Successful
....File Upload Complete
```

2. Copy a running configuration file to the SCP server.

```
device# copy running-config  scp 10.20.1.1 icx-78-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-78-startup
```

2. Copy a running configuration file from the SCP server.

```
device# copy scp running-config 10.20.1.1 icx-78-run
```

Secure Copy with SSH2

Secure Copy (SCP) uses security built into SSH to transfer image and configuration files to and from the device. SCP automatically uses the authentication methods, encryption algorithm, and data compression level configured for SSH. For example, if password authentication is enabled for SSH, the user is prompted for a user name and password before SCP allows a file to be transferred. No additional configuration is required for SCP on top of SSH.

You can use SCP to copy files on the RUCKUS device, including the startup configuration and running configuration files, to or from an SCP-enabled remote host.

Enabling and Disabling SCP

SCP is enabled by default and can be disabled.

NOTE
If you disable SSH, SCP is also disabled.

To disable SCP, enter the following command.

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

To re-enable SCP, enter the following command.

```
device# configure terminal
device(config)# ip ssh scp enable
```

Secure Copy Configuration Notes

- You may see the error "protocol error: filename does not match request" when attempting to perform an SCP copy of the following types of files: running-config, startup-config, ICX software image. This may indicate that you did not include the file extension in the source filename. Either re-enter the command and include the filename extension, or disable SCP strict filename checking feature using the -t option.
- Certain SCP client options, including -p and -r, are ignored by the SCP server on the RUCKUS device. If an option is ignored, the client is notified.

ACLs

- Layer 3 ACL Overview 85
- IPv4 ACLs 89
- IPv6 ACLs 100
- Applying an ACL to a LAG Interface 107
- Applying ACLs to VLANs 107
- ACL Logging 108
- ACL Statistics 110
- ACL Accounting 111
- Adding a Comment for an Entry in an ACL 114
- Sequence-based ACL Editing 116
- Displaying TCAM Information for ACLs 118

Layer 3 ACL Overview

Layer 3 (IPv4 and IPv6) access control lists (ACLs) permit or deny packets according to rules included in the ACLs.

NOTE
ACL configuration has changed significantly in FastIron release 08.0.95. Refer to the *RUCKUS FastIron Software Upgrade Guide* for additional information on the changes.

NOTE
ACLs cannot be configured from the web in FastIron release 08.0.95.

When a packet is received or sent, the device compares its header fields against the rules in applied ACLs. This comparison is done sequentially, in the order the rules are entered or on the sequence numbers you specify. Based on the comparison, the device either forwards or drops the packet.

ACLs include the following benefits:

- Providing security and traffic management
- Monitoring network and user traffic
- Saving network resources by classifying traffic
- Protecting against Denial of Service (DoS) attacks
- Reducing debug output

Because applied ACLs are programmed into the Content Addressable Memory (CAM), packets are permitted or denied in the hardware, without sending the packets to the CPU for processing.

Layer 3 ACLs are implemented using the following flow:

1. Create the ACL, using the **ip access-list** or **ipv6 access-list** command.
2. Define permit and deny rules, using the **deny** and **permit** commands. (A sequence number is assigned automatically to each rule, based on the order of entry.)
3. Apply the ACL to one or more interfaces or VLANs, using the relevant command:
   - IPv4: **ip access-group**
   - IPv6: **ipv6 access-group**

Layer 3 ACLs are supported on all ICX device physical interface types and LAGs.

Named ACLs and numbered ACLs are supported for IPv4 ACLs. IPv6 ACLs are named. Named ACLs must begin with an alphabetical character and can contain up to 47 alphanumeric characters.

Special characters, such as <, >, and &, are not allowed, except for double quotes.

NOTE
For Layer 2 filtering, refer to MAC ACLs on page 125.

NOTE
For ACLs under Flexible authentication, refer to Dynamic ACLs in Authentication on page 196.

ACL Scaling

For each ACL type, there is a software limit to the number of ACLs supported. The maximum number of ACL rules supported also varies with the device. The following table contains scaling information for all ICX devices.

TABLE 13 ACL Rule Scaling Limits

Security Feature	ICX 8200	ICX 7850	ICX 7650	ICX 7550
Software Scale				
Maximum configurable standard numbered IPv4 ACLs	99	99	99	99
Maximum configurable extended numbered IPv4 ACLs	100	100	100	100
Maximum configurable standard named IPv4 ACLs	600	600	600	600
Maximum configurable extended named IPv4 ACLs	600	600	600	600
Maximum configurable IPv6 ACLs	600	600	600	600
Maximum configurable MAC ACLs	3,072	3,072	3,072	3,072
Maximum configurable filters per IP ACL (same as system max parameter ip-filter-port)	2,048	2,048	2,048	2,048
Maximum configurable IP filters (IPv4 and IPv6) for the entire stack across all ACLs (same as system max parameter ip-filter-sys)	8,192	8,192	8,192	8,192
Maximum configurable filters per MAC ACL (same as system max parameter mac-filter-port)	256	256	256	256
Maximum configurable MAC filters for the entire stack across all ACLs (same as system max parameter mac-filter-sys)	3,072	3,072	3,072	3,072
DHCP Snooping/IPSG/DAI				
Max DHCP Snooping entries	32,768	32,768	32,768	32,768

TABLE 13 ACL Rule Scaling Limits (continued)

Security Feature	ICX 8200	ICX 7850	ICX 7650	ICX 7550
Max number of IP Source Guard entries per device	1,024[1]	1,536	4,096	2,048
Max Dynamic ARP Inspection entries	32,768	32,768	32,768	32,768
Max Static ARP Inspection entries	6,000	6,000	6,000	6,000
Maximum configurable filters per SG ACL	8	8	8	8
DHCPv6 Snooping/ND Inspection				
Max DHCPv6 Snooping entries	32,768	32,768	32,768	16,384
Max IPv6 Neighbor Discovery Inspection entries	32,768	32,768	32,768	16,384
Max Static IPv6 Neighbor Discovery Inspection entries	6,000	6,000	6,000	6,000
Max IPv6 Source Guard Entries per device	512	1,536	2,048	2,048
Hardware Scale				
IPv4 ingress TCAM rules per device (IPv4 ACL/IPSG)	1024[2]	1,536	4,096	2,048
IPv6 ingress TCAM rules per device	512[3]	1,536	2,048	2,048
IPv4 Egress TCAM rules per device	128	512	256	256
IPv6 Egress TCAM rules per device	128	512	256	256
L2 Ingress TCAM rules per device	512	1,536	1,536	2,048

ACL Scaling Considerations

Keep the following items in mind when configuring ACLs.

- All platforms consume 1 TCAM space by default for egress IPv4 and IPv6 groups, which reduces the space available for rules by 1 for IPv4 and IPv6 egress ACLs.
- On ICX 7550 and ICX 7850 devices, when an egress ACL is applied to a VLAN, every ACL rule, including each default rule, is programmed as 2 entries.
- By default, TCAM reserves 5 entries for an IPv4 ingress ACL group and 30 entries for a Layer 2 (MAC) ingress ACL on all ICX platforms.
- Use the **show access-list tcam usage unit** *id* command to review hardware usage before binding an ACL.

[1] The maximum number of IPv4 Source Guard Entries per ICX 8200-C08PF device is 512. The maximum number of IPv6 Source Guard Entries per ICX 8200-C08PF device is 256.
[2] ICX 8200-C08PF devices support 512 IPv4 ingress TCAM rules (IPv4 ACL) per device.
[3] ICX 8200-C08PF devices support 256 IPv6 ingress TCAM rules (IPv6 ACL) per device.

Example:

```
device(config)# show access-list tcam usage unit 3
UnitId Region Group Id   Direction       Type               : Allocated  Total      Free
------ ------ --------   ---------       ----               : ---------  -----      ----
3      0      1          Pre-Ingres      L2_IPv4 FIlters    : 2          256        254
3      0      2          Pre-Ingres      VCAP_MISC          : 8          512        504
3      0      3          Ingress         IPv4 Filters       : 5          2048       2043
3      0      4          Ingress         IPv6 Filters       : 0          1280       1280
3      0      5          Ingress         L2 Filters         : 30         2048       2018
3      0      6          Ingress         ICAP All Combo     : 51         1024       973
3      0      7          Egress          IPv4 Filters       : 1          256        255
3      0      8          Egress          IPv6 Filters       : 1          256        255
3      0      9          Egress          L2 Filters         : 3          256        253
```

- Published scale numbers are one dimensional. If IPv4, IPv6, and MAC ACLs are cofigured on the same device, one-dimensional scaling numbers do not apply to the combined ACLs.
- On an ICX 7850 device, if you migrate to FastIron 08.0.95 or a later release from a FastIron 08.0.92 configuration that contains an IPv4 egress ACL applied to a virtual interface, the 2 TCAM rules originally programmed for the ACL (one ACL rule and one implicit deny rule), are programmed as 4 TCAM rules in the target release configuration, where the ACL will be applied at the VLAN level; that is, 2 rules for the ACL and 2 rules for the implicit deny rule.
- On an ICX 7850 device, if you migrate from FastIron 08.0.92 to FastIron 08.0.95 or a later release, the rules created for an IPv6 egress ACL applied to a virtual interface multiply. For example, if you created the original IPv6 egress ACL with one rule, the ACL is programmed as 4 rules in TCAM for the FastIron 08.0.92 configuration; that is, 1 IPv6 ACL rule and 3 implicit rules. In the resulting configuration for the target release, the IPv6 ACL is applied at the VLAN level, and a total of 8 rules will be created in TCAM; that is, 2 ACL rules and 6 implicit rules.

  NOTE
  On ICX 7850 devices, there is no change in scale when you apply an egress ACL on a physical interface.

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 or VLAN, the default action for that port or VLAN is to deny all traffic that is not explicitly permitted. Given these defaults, follow these guidelines for configuring and applying ACLs:

- If you want to control access, configure ACLs consisting of permit entries for the access you want to permit. The ACLs implicitly deny all other access.
- If you want to secure access in environments with many users, you may want to configure ACLs that consist of explicit deny entries and then add an entry to permit all access to the end of each ACL so that the software permits packets that are not denied by the deny entries.

Because by default traffic is restricted unless explicitly allowed once an ACL is applied to an interface or VLAN, an ACL can block packets switched between members of the same VLAN or subnet over a virtual interface with an ACL applied to the VLAN or a port in it. To overcome this issue, you must add explicit rules to the ACL allowing traffic from the sources you want to include.

Consider the following example.

```
vlan 200 by port
 untagged ethe 1/1/1 to 1/1/2
 !
interface ve 200
 ip address 192.168.1.1 255.255.255.0
!
ip access-list extended acl101
permit ip host 1.1.1.1 host 2.2.2.2
!
ip access-list extended acl102
```

```
permit ip host 1.1.1.1 host 2.2.2.2
permit ip 192.168.1.0 0.0.0.255 192.168.1.0 0.0.0.255
!

host1 (192.168.1.10/24) is connected to 1/1/1 and host2 (192.168.1.11/24) is connected to 1/1/2.
```

If access-list acl101 is applied to VLAN 200, the hosts in it will not be able to communicate with each other. This is because acl101 includes an implicit deny filter at the end by default.

However, if access-list acl102 is applied to VLAN 200, the hosts in it will be able to communicate with each other. This is because the second statement of the ACL explicitly permits traffic within the subnet 192.168.1.0/24, to which the hosts belong.

How Hardware-based ACLs Work

When you bind an ACL to inbound or outbound traffic on an interface or VLAN, 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 or VLAN matches an ACL rule in the Layer 4 CAM, the device permits or denies the packet according to the ACL.
- If a packet does not match an ACL rule, the packet is dropped because the default action on an interface or VLAN that has ACLs is to deny the packet.

How Fragmented Packets Are Processed

Fragments are processed by hardware-based ACLs by default in the following manner:

- The first fragment of a packet is permitted or denied using the ACLs. The first fragment is handled the same way as non-fragmented packets, because the first fragment contains the Layer 4 source and destination application port numbers. The device uses the Layer 4 CAM entry if one is programmed, or applies the interface's ACL entries to the packet and permits or denies the packet according to the first matching ACL.
- Other fragments of the same packet are subject to a rule only if there is no Layer 4 information in the rule or in any preceding rules.

The fragments are forwarded even if the first fragment, which contains the Layer 4 information, was denied. Generally, denying the first fragment of a packet is sufficient, because a transaction cannot be completed without the entire packet.

For tighter control, you can configure the port to drop all packet fragments. Refer to Enabling Strict Control of ACL Filtering of Fragmented Packets on page 93.

IPv4 ACLs

IPv4 ACLs permit or deny IPv4 packets according to rules included in the ACLs.

Regarding the range of filtering options, there are two types of IPv4 ACLs:

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

The following points apply to named ACLs:

- An ACL name must begin with an alphabetical character followed by alphanumeric characters.
- The maximum length of an ACL name is 47 characters.
- An ACL name cannot contain special characters such as <, >, and &.

  Double quotes are allowed in the following release streams from the specified patch release:

  - FastIron 08.0.95j and later
  - FastIron 09.0.10e and later
  - FastIron 10.0.00a and later
- The ACL name cannot be 'test'.

IPv4 ACL Configuration Guidelines

- Flow-based ACLs are not supported on RUCKUS ICX devices.
- Inbound ACLs apply to all traffic, including control traffic. By default, outbound ACLs are not applied to traffic generated by the CPU. To enable the application of outbound ACLs to CPU traffic, use the **enable egress-acl-on-cpu-traffic** command. Refer to Applying Egress ACLs to Control (CPU) Traffic on page 93 for details.
- The RUCKUS ICX ACL implementation supports only one ACL per port or VLAN. The ACL can contain multiple entries (rules). For example, hardware-based ACLs do not support ACL101 and ACL102 on port 1/1/1 or VLAN 100, but hardware-based ACLs do support ACL101 containing multiple entries. If a user tries to apply a second ACL on a port or VLAN, the second ACL will replace the current ACL.
- Inbound ACLs and outbound ACLs can be configured on the same port or VLAN.
- By default, the first fragment of a fragmented packet received by a FastIron device is permitted or denied using the ACLs, but subsequent fragments of the same packet are forwarded in hardware. Generally, denying the first fragment of a packet is sufficient because a transaction cannot be completed without the entire packet.
- ACLs are supported on member ports of a VLAN on which DHCP snooping and Dynamic ARP Inspection (DAI) are enabled. However, IP source guard and ACLs are not supported together at the port level, at the VLAN level, or across levels.
- Outbound ACLs cannot be configured through a RADIUS server as dynamic or user-based ACLs. However, outbound ACLs can still be configured with MAC authentication or 802.1X authentication enabled, as they are configured in different directions.
- On all platforms, ACL support for switched inbound and outbound traffic is enabled by default.
- IPv4 ACLs are supported on tagged ports.
- An outbound IPv4 ACL with a rule pertaining to ESP protocol does not match ESP packets with IPv4 headers on ICX 7550 or ICX 7850 devices.
- You can apply an ACL to a port that has TCP SYN attack protection or ICMP smurf attack protection, or both, enabled.
- When forming LAGs, make sure that no ACLs are currently applied to the relevant physical interfaces.
- When using host names in ACL filter configurations, a TCAM entry will be programmed with the first IP address resolved through DNS service. Any further updates to DNS entries do not automatically update TCAM.
- It is not possible to configure conflicting ACL filters. When an ACL filter is configured, if the sequence number already exists, or the sequence number is not specified explicitly and all the filter parameters match with an existing filter but the action does not match, the following error message is displayed:

  Error: ACL operation failed for ACL ipv4-test1 since following conflicting filter entry already exists. Please use explicit sequence number to override this error.
- ACL mirroring is not supported for outbound ACLs.

Creating and Applying a Standard IPv4 ACL

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

Complete the following steps to create a standard ACL.

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

```
device# configure terminal
```

2. Enter the **ip access-list standard** command followed by a name or ID number to create the ACL and enter ACL configuration sub-mode. An ID number must be all numeric and be in the range 1 through 99. If you use a name, the name must begin with an alphabetical character and be no more than 47 characters long.

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

3. For each rule, enter the **deny** or **permit** command followed by needed parameters. As an option, you may specify the sequence number followed by a permit or deny statement. Otherwise, the sequence numbers will be assigned automatically in the order of statement entry in increments of 10.

```
device(config-std-ipacl_ip_stan_test)# deny host 10.157.22.26 log
device(config-std-ipacl_ip_stan_test)# deny 10.157.29.12 log
device(config-std-ipacl_ip_stan_test)# deny host IPHost1 log
device(config-std-ipacl_ip_stan_test)# permit any
```

4. Apply the ACL you created to the appropriate interface or VLAN and specify direction. If desired, include the **logging enable** option to log matched statements that contain the keyword **log**.

```
device(config-std-ipacl_ip_stan_test)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# ip access-group ip_stan_test in logging enable
```

The following example configures an ACL to deny packets from three source IP addresses being received on port 1/1/1. The last rule permits all packets not explicitly denied by the first three ACL entries. (Otherwise, the implicit action is "deny".) In the example, the ACL is applied to the port along with the keywords **logging enable**. As a result, all deny actions, which include the keyword **log**, are logged.

```
device# configure terminal
device(config)# ip access-list standard ip_stan_test
device(config-std-ipacl_ip_stan_test)# deny host 10.157.22.26 log
device(config-std-ipacl_ip_stan_test)# deny 10.157.29.12 log
device(config-std-ipacl_ip_stan_test)# deny host IPHost1 log
device(config-std-ipacl_ip_stan_test)# permit any
device(config-std-ipacl_ip_stan_test)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# ip access-group ip_stan_test in logging enable
device(config-if-e1000-1/1/1)# exit
device(config)#
```

The following example is the result of entering **show access-list all** for the previously configured ACL.

```
device# show access-list all

Standard IP access list ip_stan_test: 4 entries
10: deny host 10.157.22.26 log
20: deny host 10.157.29.12 log
30: deny host IPHost1 log
40: permit any
```

The following example includes remarks preceding each rule. For more information on adding remarks, refer to Adding a Comment for an Entry in an ACL on page 114.

```
device(config)# ip access-list standard 10
device(config-std-ipacl-10)# remark server-lab10-backup
device(config-std-ipacl-10)# permit host 192.168.100.14
device(config-std-ipacl-10)# remark clients-lab10
device(config-std-ipacl-10)# permit 192.168.100.0 0.0.0.248
device(config-std-ipacl-10)# exit
device(config)#
```

IPv4 Extended ACL Traffic Filtering Criteria

Extended ACLs can permit or deny traffic based on the following criteria:

- IPv4 source address
- IPv4 destination address
- Protocol names
- Protocol numbers
- Layer 4 port numbers

For more information, refer to the **ip access-list** command in the *RUCKUS FastIron Command Reference*.

Creating and Applying an Extended IPv4 ACL

Complete the following steps to create an extended IPv4 ACL.

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

   ```
   device# configure terminal
   ```

2. Enter the **ip access-list extended** command followed by a name or ID to create the ACL and enter ACL configuration sub-mode. An ID number must be all numeric and be in the range 100 through 199. If you use a name, it must begin with an alphabetical character and contain no more than 47 characters.

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

3. For each rule, enter the **permit** or **deny** command, specifying the needed parameters. As an option, you can specify a sequence number followed by a permit or deny statement. Otherwise, the sequence numbers will be assigned automatically in the order of statement entry in increments of 10.

   ```
   device(config-ext-ipacl-ip_ext_test)# deny tcp host 10.157.22.26 any eq telnet log
   device(config-ext-ipacl-ip_ext_test)# permit ip any any
   ```

4. Apply the ACL you created to the needed interfaces or VLANs using the **ip access-group** command and specify the direction. If desired, include the **logging enable** option to log matched statements that contain the keyword **log**.

   ```
   device(config-ext-ipacl-ip_ext_test)# interface ethernet 1/1/2
   device(config-if-e1000-1/1/2)# ip access-group ip_ext_test in logging enable
   device(config-if-e1000-1/1/2)# exit
   device(config)#
   ```

The following example includes remarks preceding each rule.

```
device# configure terminal
device(config)# ip access-list extended ip_ext_test
device(config-ext-ipacl-ip_ext_test)# remark Permits ICMP traffic from 10.157.22.x to 10.157.21.x:
device(config-ext-ipacl-ip_ext_test)# permit icmp 10.157.22.0/24 10.157.21.0/24
device(config-ext-ipacl-ip_ext_test)# remark Denies IGMP traffic from "rkwong" to 10.157.21.x:
device(config-ext-ipacl-ip_ext_test)# deny igmp host rkwong 10.157.21.0/24 log
device(config-ext-ipacl-ip_ext_test)# remark Denies IGRP traffic from "rkwong" to 10.157.21.x:
device(config-ext-ipacl-ip_ext_test)# deny igrp 10.157.21.0/24 host rkwong log
device(config-ext-ipacl-ip_ext_test)# remark Denies IPv4 traffic from 10.157.21.100 to 10.157.22.1, with
logging:
device(config-ext-ipacl-ip_ext_test)# deny ip host 10.157.21.100 host 10.157.22.1 log
device(config-ext-ipacl-ip_ext_test)# remark Denies all OSPF traffic, with logging:
device(config-ext-ipacl-ip_ext_test)# deny ospf any any log
device(config-ext-ipacl-ip_ext_test)# remark Permits traffic not explicitly denied by the previous rules:
device(config-ext-ipacl-ip_ext_test)# permit ip any any
```

Applying Egress ACLs to Control (CPU) Traffic

With some exceptions, outbound ACLs are not by default applied to traffic generated by the CPU. To apply outbound ACLs to CPU traffic, use the **enable egress-acl-on-cpu-traffic** command.

CPU traffic on standby and member units in ICX 7550 and ICX 7850 stacks is, by default, subjected to outbound ACL filtering. In contrast, ICX 7550 and ICX 7850 standalone devices and stack active-controllers have the same default behavior as other platforms and do not apply outbound ACLs to CPU traffic.

Enabling Strict Control of ACL Filtering of Fragmented Packets

For strict control of ACL filtering of fragmented packets received on an interface, you can configure the interface to drop all packet fragments. To do so, apply the **ip access-group frag deny** command to the interface. In the following example, the ACL is applied to port 1/1/1.

```
device# configure terminal
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# ip access-group frag deny
```

This option begins dropping all fragments received by the port as soon as you enter the command. The option is especially useful if the port is receiving an unusually high rate of fragments, which can indicate a hacker attack.

NOTE
The **ip access-group frag deny** command is not supported on LAG interfaces.

Filtering on IP Precedence and ToS Values

The following example configures an extended IP ACL that matches based on IP precedence.

```
device(config)# ip access-list extended acl103
device(config-ext-ipacl-acl103)# deny tcp 10.157.21.0/24 10.157.22.0/24 precedence internet
device(config-ext-ipacl-acl103)# deny tcp 10.157.21.0/24 eq ftp 10.157.22.0/24 precedence 6
device(config-ext-ipacl-acl103)# permit ip any any
```

The first entry in this ACL denies TCP traffic from the 10.157.21.x network to the 10.157.22.x network if the traffic has the IP precedence option "internet" (equivalent to "6").

The second entry denies all FTP traffic from the 10.157.21.x network to the 10.157.22.x network if the traffic has the IP precedence value "6" (equivalent to "internet").

The third entry permits all packets that are not explicitly denied by the other entries. Without this entry, the ACL would deny all incoming or outgoing IP traffic on the ports to which you assign the ACL.

The following example configures an IP ACL that matches based on ToS.

```
device(config)# ip access-list extended acl104
device(config-ext-ipacl-acl104)# deny tcp 10.157.21.0/24 10.157.22.0/24 tos normal
device(config-ext-ipacl-acl104)# deny tcp 10.157.21.0/24 eq ftp 10.157.22.0/24 tos 13
device(config-ext-ipacl-acl104)# permit ip any any
```

The first entry in this IP ACL denies TCP traffic from the 10.157.21.x network to the 10.157.22.x network if the traffic has the IP ToS option "normal" (equivalent to "0").

The second entry denies all FTP traffic from the 10.157.21.x network to the 10.157.22.x network if the traffic has the IP ToS value "13" (equivalent to "max-throughput", "min-delay", and "min-monetary-cost").

The third entry permits all packets that are not explicitly denied by the other entries. Without this entry, the ACL would deny all incoming or outgoing IP traffic on the ports to which you assign the ACL.

ACLs to Filter ARP Packets

NOTE
Using ACLs to filter ARP packets is not applicable to outbound traffic.

You can use ACLs 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 the normal filtering provided by ACLs.

When a RUCKUS device receives an ARP request, the source MAC and IP addresses are stored in the device ARP table. A new record in the ARP table overwrites existing records that contain the same IP address. This behavior can cause a condition called ARP hijacking when two hosts with the same IP address try to send an ARP request to the RUCKUS device.

Normally, ARP hijacking is not a problem because IP assignments are done dynamically; however, in some cases, ARP hijacking can occur, such as when a configuration allows a router interface to share the IP address of another router interface. Because multiple VLANs and the router interfaces that are associated with each of the VLANs share the same IP segment, it is possible for two hosts in two different VLANs to fight for the same IP address in that segment. ARP filtering using ACLs protects an IP host record in the ARP table from being overwritten by a hijacking host. Using ACLs to filter ARP requests checks the source IP address in the received ARP packet. Only packets with the permitted IP address will be allowed to be written in the ARP table. Other packets are dropped.

Configuration Considerations for Filtering ARP Packets

- Filtering ARP packets is available on physical interfaces and virtual routing interfaces. It is supported on the following physical interface types: Ethernet and LAG ports.
- IPv4 standard numbered or IPv4 extended numbered ACLs can be used. If any other type of ACL is used, an error is displayed.

Configuring ACLs for ARP Filtering

To configure ACLs for ARP filtering, enter commands such as the following.

```
device(config)# ip access-list extended 101
device(config-ext-ipacl-101)# permit ip host 192.168.2.2 any
device(config-ext-ipacl-101)# exit
device(config)# ip access-list extended 102
device(config-ext-ipacl-102)# permit ip host 192.168.2.3 any
device(config-ext-ipacl-102)# exit
device(config)# ip access-list extended 103
device(config-ext-ipacl-103)# permit ip host 192.168.2.4 any
device(config-ext-ipacl-103)# exit
device(config)# vlan 2
device(config-vlan-2)# tagged ethernet 1/1/1 to 1/1/2
device(config-vlan-2)# vlan 3
```

```
device(config-vlan-3)# tagged ethernet 1/1/1 to 1/1/2
device(config-vlan-3)# vlan 4
device(config-vlan-4)# tagged ethernet 1/1/1 to 1/1/2
device(config-vlan-4)# vlan 2
device(config-vlan-2)# ip access-group 101 in
device(config-vlan-2)# ip address 192.168.2.1/24
device(config-vlan-2)# interface ve 2
device(config-vif-2)# ip use-acl-on-arp 103
device(config-vif-2)# vlan 3
device(config-vlan-3)# ip access-group 102 in
device(config-vlan-3)# interface ve 3
device(config-vif-3)# ip use-acl-on-arp 103
device(config-vif-3)# vlan 4
device(config-vlan-4)# interface ve 4
device(config-vif-4)# ip use-acl-on-arp 103
device(config-vif-4)# exit
device(config)#
```

When the **ip use-acl-on-arp** command is configured, the ARP module checks the source IP address of the ARP request packets received on the interface. It then applies the specified ACL policies to the packet. Only the packet with the IP address that the ACL permits will be written in the ARP table. The packets that are not permitted will be dropped.

The ACL ID identifies the standard or extended IPv4 ACL that will be used to filter the packet. Only the source IP address, in the case of a standard ACL, or the source and destination IP addresses, in the case of an extended ACL, will be used to filter the ARP packet. Enter an ACL number to explicitly specify the ACL to be used for filtering. In the example, the **ip use-acl-on-arp 103** command specifies ACL 103 to be used as the filter.

ARP requests will not be filtered by ACLs if an ACL ID is specified for the **ip use-acl-on-arp** command, but no IP address or "any any" filtering criteria have been defined under the ACL name.

Displaying ACL Filters for ARP

To determine which ACLs have been configured to filter ARP requests, enter the **show acl-on-arp** command.

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

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

The Filter Count column in the command output shows how many ARP packets have been dropped on the interface since the last time the count was cleared.

Clearing the Filter Count

To clear the filter count for all interfaces on the device, enter the **clear acl-on-arp** command.

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

The command resets the filter count to zero on all device interfaces.

QoS Options for IP ACLs

ACL Quality of Service (QoS) parameters enable you to perform QoS for packets that match the ACLs.

Using an ACL to perform QoS is an alternative to directly setting the internal forwarding priority based on incoming port, VLAN membership, and so on. The following QoS ACL options are supported:

- **dscp-cos mapping**: This option is similar to the **dscp-matching** command. This option maps the DSCP value in incoming packets to a hardware table that provides mapping of each of the 0 through 63 DSCP values and distributes them among eight traffic classes (internal priorities) and eight 802.1p priorities.
- **dscp-marking**: Marks the DSCP value in the outgoing packet with the value you specify.
- **internal-priority-marking** and **802.1p-priority-marking**: Supported with the DSCP marking option, these commands assign traffic that matches the ACL to a hardware forwarding queue (**internal-priority-marking**) and remark the packets that match the ACL with the 802.1p priority (**802.1p-priority-marking**).
- **dscp-matching**: Matches on the packet DSCP value. This option does not change the packet-forwarding priority through the device or mark the packet.
- **802.1p-priority-matching**: Inspects the 802.1p bit in the ACL that can be used with adaptive rate limiting.

  NOTE
  The **internal-priority-marking** option is not supported for egress ACLs.

  NOTE
  For more information, refer to **sequence (permit|deny in extended IPv4 ACLs)** in the *RUCKUS FastIron Command Reference*.

Configuration Notes for QoS Options

ICX devices do not support marking and prioritization simultaneously with the same rule and do not support DSCP CoS mapping at all. Instead, you must create two separate rules. In other words, you can mark a rule with DSCP or 802.1p information, or you can prioritize a rule based on DSCP or 802.1p information. You can enable only one of the following ACL options per rule:

- **802.1p-priority-marking**
- **dscp-marking**
- **internal-priority-marking**

  NOTE
  The **internal-priority-marking** option is not supported for egress ACLs.

For example, any one of the following commands is supported:

```
device(config)# ip access-list extended acl101
device(config-ext-ipacl-acl101)# permit ip any any dscp-marking 43
```

or

```
device(config)# ip access-list extended acl101
device(config-ext-ipacl-acl101)# permit ip any any internal-priority-marking 6
```

or

```
device(config)# ip access-list extended acl101
device(config-ext-ipacl-acl101)# permit ip any any dscp-marking 43 802.1p-priority-marking 4 internal-
priority-marking 6
```

Using a Combined ACL for 802.1p Marking

ICX devices support a simple method for assigning an 802.1p priority value to packets without affecting the actual packet or the DSCP.

When you specify 802.1p priority marking values directly, internal priority marking is optional.

NOTE

Using a combined ACL for 802.1p marking is not applicable to outbound traffic.

The following example, for any IP protocol, specifies 802.1p priority marking "1".

```
device(config)# ip access-list extended acl104
device(config-ext-ipacl-acl104)# permit ip any any 802.1p-priority-marking 1
```

The following example, for any IP protocol, specifies 802.1p priority marking "1" and internal priority marking "5".

```
device(config)# ip access-list extended acl104
device(config-ext-ipacl-acl104)# permit ip any any 802.1p-priority-marking 1 internal-priority-marking 5
```

The following example, for TCP, specifies 802.1p priority marking "1".

```
device(config)# ip access-list extended acl105
device(config-ext-ipacl-acl105)# permit tcp any any 802.1p-priority-marking 1
```

The following example, for TCP, specifies 802.1p priority marking "1" and internal priority marking "5".

```
device(config)# ip access-list extended acl105
device(config-ext-ipacl-acl105)# permit tcp any any 802.1p-priority-marking 1 internal-priority-marking 5
```

The following example, for UDP, specifies 802.1p priority marking "1".

```
device(config)# ip access-list extended acl105
device(config-ext-ipacl-acl105)# permit udp any any 802.1p-priority-marking 1
```

The following example, for UDP, specifies 802.1p priority marking "1" and internal priority marking "5".

```
device(config)# ip access-list extended acl105
device(config-ext-ipacl-acl105)# permit udp any any 802.1p-priority-marking 1 internal-priority-marking 5
```

In each of the preceding pairs of examples, the first command does not specify the internal-priority value, which means it maintains a default value of 1 (equal to the 802.1p value). In the second command, the internal-priority value is configured to 5.

Configuring QoS Priority for a VLAN

Use this procedure to configure QoS priority for a VLAN.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Check the target VLAN configuration.

   ```
   device(config)# show running-config vlan 100
   vlan 100 by port
    tagged ethe 1/1/3 to 1/1/5
   !
   !
   ```

   In this example, VLAN 100 is used.

3. Enter VLAN configuration mode for the target VLAN.

   ```
   device(config)# vlan 100
   ```

4. Attach a router interface to the target VLAN.

```
device(config-vlan-100)# interface ve 100
```

5. Verify the VLAN configuration.

```
device(config-vlan-100)# show running-config vlan 100
vlan 100 by port
 tagged ethe 1/1/3 to 1/1/5
 interface ve 100
!
```

6. Return to global configuration mode.

```
device(config-vlan-100)# exit
```

7. Create an extended ACL.

```
device(config)# ip access-list extended acl-remark
```

In this example, the extended ACL is named "acl-remark".

8. Configure a rule in the ACL with the required qualifiers and the internal priority that you want.

```
device(config-ext-ipacl-acl-remark)# permit ip any any 802.1p-and-internal-marking 5
```

In this example, QoS priority 5 is used.

9. Verify the ACL configuration.

```
device(config-ext-ipacl-acl-remark)# show ip access-lists all

Extended IP access list  acl-remark : 1 entry
permit ip any any  802.1p-and-internal-marking 5
```

10. Go to VLAN configuration mode.

```
device(config-ext-ipacl-acl-remark)# vlan 100
```

11. Apply the ACL to VLAN 100.

```
device(config-vlan-100)# ip access-group acl-remark in
```

QoS Priority for a VLAN Configuration Example

```
device# configure terminal
device(config)# show running-config vlan 100
!
device(config)# vlan 100
!
device(config-vlan-100))# show running-config
device(config-vlan-100))# exit
device(config)# ip access-list extended acl-remark
device(config-ext-ipacl-acl-remark)# permit ip any any 802.1p-and-internal-marking 5
device(config-ext-ipacl-acl-remark)# show ip access-lists all
!
device(config-ext-ipacl-acl-remark)# vlan 100
device(config-vlan-100)# ip access-group acl-remark in
```

DSCP Matching

The **dscp-matching** option matches on the packet DSCP value. This option does not change the packet forwarding priority through the device or mark the packet.

To configure an ACL that matches on a packet with DSCP value 29, enter a command such as the following.

```
device(config)# ip access-list extended acl112
device(config-ext-ipacl-acl112)# permit ip 1 0.1.1.0 0.0.0.255 10.2.2.x 0.0.0.255 dscp-matching 29
```

ACL-based Rate Limiting

ACL-based rate limiting provides the facility to limit the rate for inbound IP traffic that matches the permit conditions in extended IP ACLs.

> **NOTE**
> For implementation details, refer to "Rate Limiting and Rate Shaping" in the *RUCKUS FastIron QoS and Traffic Management Configuration Guide*.

ACLs to Control Multicast Features

You can use ACLs to control multicast features in the following ways:

- Limit the number of multicast groups that are covered by a static rendezvous point (RP).
- Control for which multicast groups candidate RPs send advertisement messages to bootstrap routers.
- Identify which multicast group packets will be forwarded or blocked on an interface.

Displaying IPv4 ACL Information

Use the following commands to display information about IPv4 ACLs:

- **show ip access-lists**
- **show access-list accounting**
- **show running-config access-list**

The following example shows that four IPv4 ACLs are configured on the device.

```
device# show ip access-lists brief

Standard IP access list 1: 1 entries

Extended IP access list 102: 0 entries

Extended IP access list 104: 0 entries

Standard IP access list 11: 100 entries
```

The following example shows the statements defined for the IPv4 ACLs configured on the device.

```
device# show ip access-lists

Extended IP access list 101: 1 entries
10: permit ip 1.1.1.1 1.1.1.1 any mirror

Standard IP access list test1: 2 entries
10: deny host 10.157.22.26 log
20: permit any

Extended IP access list icmp_pass: 1 entries
10: permit icmp any any
```

The following example shows all IPv4 ACLs with the interface to which they are bound and their active direction.

```
device# show ip access-lists bindings
ACL NAME  TARGET                DIRECTION
========= =======               ==========
Acl1      eth 3/1/2             in
Acl2      lag 1                 out
Acl1      ve 10                 in
Acl2      vlan 20               in
Acl3      vlan 20:lag 2         in
Acl4      vlan 20:eth 2/1/1     in
Acl5      ve 10:lag 3           out
Acl5      ve 10:eth 1/1/1       out
Acl7      cpu-ports:unit-active in
```

The following example displays a summary of ACLs that are active on VLAN 333.

```
device# show access-list accounting vlan 333 in brief
ACL Accounting Table
==========================================================================================================
==============================================
UnitID: 1 ACL Name: mirror_acl_l2     HitCnt: 0   ByteCnt: 0
UnitID: 1 ACL Name: mirror_acl_ipv4   HitCnt: 0   ByteCnt: 0
UnitID: 1 ACL Name: mirror_acl_ipv6   HitCnt: 0   ByteCnt: 0
UnitID: 2 ACL Name: mirror_acl_l2     HitCnt: 0   ByteCnt: 0
UnitID: 2 ACL Name: mirror_acl_ipv4   HitCnt: 0   ByteCnt: 0
UnitID: 2 ACL Name: mirror_acl_ipv6   HitCnt: 0   ByteCnt: 0
UnitID: 3 ACL Name: mirror_acl_l2     HitCnt: 0   ByteCnt: 0
UnitID: 3 ACL Name: mirror_acl_ipv4   HitCnt: 0   ByteCnt: 0
UnitID: 3 ACL Name: mirror_acl_ipv6   HitCnt: 0   ByteCnt: 0
```

The following example displays IPv4 ACLs present in the running configuration.

```
device# show running-config access-list ip
ip access-list extended acl1
sequence 10 permit ip host 192.168.10.11 any
sequence 20 deny ip host 192.168.10.12 any
sequence 30 deny udp any any log
sequence 40 permit tcp any any mirror
!
```

IPv6 ACLs

FastIron devices support IPv6 Access Control Lists (ACLs) for inbound and outbound traffic filtering.

For information on IPV6 ACL limits, refer to ACL Scaling on page 86.

IPv6 ACL Traffic Filtering Criteria

The RUCKUS implementation of IPv6 ACLs enables traffic filtering based on the following information:

- IPv6 protocol
- Source IPv6 address
- Destination IPv6 address
- IPv6 message type
- Source TCP or UDP port (if the IPv6 protocol is TCP or UDP)
- Destination TCP or UDP port (if the IPv6 protocol is TCP or UDP)
- IPv6 packet fragments

NOTE
RUCKUS ICX 8200 switches include the first IPv6 packet fragments in ACL rule matching. In contrast, other ICX devices include non-first fragments in rule matching.

NOTE
When setting the ACL rule to filter specific ICMP packets, the IPv6 ACL mirroring option is not supported. As a result, the **permit icmp any any echo-request mirror** command cannot be used.

IPv6 Protocol Names and Numbers

The IPv6 protocol can be one of the following well-known names or any IPv6 protocol number from 0 through 255:

- Authentication Header (AH) protocol
- Encapsulating Security Payload (ESP)
- Internet Control Message Protocol (ICMP)
- Internet Protocol Version 6 (IPv6)
- Stream Control Transmission Protocol (SCTP)
- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

NOTE
TCP and UDP filters will be matched only if they are listed as the first option in the extension header.

For TCP and UDP, you also can specify a comparison operator and port name or number. For example, you can configure a policy to block web access to a specific website by denying all TCP port 80 (HTTP) packets from a specified source IPv6 address to the website IPv6 address.

IPv6 ACLs also provide support for filtering packets based on DSCP.

Default and Implicit IPv6 ACL Action

There are default actions and implicit IPv6 ACL rules.

The default action when no ACLs are configured on an interface is to permit all traffic. However, once you configure an ACL and apply it to an interface, the default action for that interface is to deny all traffic that is not explicitly permitted on the interface.

- If you want to tightly control access, configure ACLs consisting of permit rules for the access you want to permit. The ACLs implicitly deny all other access.
- If you want to secure access in environments with many users, you may want to configure ACLs that consist of explicit deny rules, with a **permit ipv6 any any** rule at the end of each ACL. You must enter **permit ipv6 any any** as the last statement in the access list if you want to permit IPv6 traffic that was not explicitly denied by the previous statements.

IPv6 ACLs have the following concluding implicit rules:

- **permit icmp any any nd-na**: Allows ICMP neighbor discovery acknowledgements.
- **permit icmp any any nd-ns**: Allows ICMP neighbor discovery solicitations.
- **deny ipv6 any any**: Denies IPv6 traffic.

NOTE
In an IPv6 ACL, if you do not override the implicit **deny ipv6 any any** rule, make sure to include rules that permit the IPv6 link-local address and the global unicast address. Otherwise, routing protocols such as OSPF will not work. To view the link-local address, use the **show ipv6 interface** command.

For example, if you want to deny ICMP neighbor discovery acknowledgment and 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 acknowledgments.

The last rule permits all packets that are not explicitly denied by the other rules. Without this rule, the ACL will deny all incoming IPv6 traffic on the ports to which you assigned the ACL.

If you want to deny all neighbor discovery messages, use the statement **deny icmp any any** in the access list. If you want to permit neighbor discovery but deny all other ICMP traffic, use the **permit icmp any any nd-na** and **permit icmp any any nd-ns** statements just before the **deny icmp** statement as shown in the following example.

```
device(config)# ipv6 access-list netw
device(config-ipv6-access-list netw)# permit icmp 2001:DB8:e0bb::/64 2001:DB8::/64
device(config-ipv6-access-list netw)# permit icmp any any nd-na
device(config-ipv6-access-list netw)# permit icmp any any nd-ns
device(config-ipv6-access-list netw)# deny icmp any any
device(config-ipv6-access-list netw)# permit ipv6 any any
```

NOTE
To configure protection against spurious neighbor discovery (ND) packets employed in Denial of Service (DoS) attacks, refer to Neighbor Discovery (ND)-Packet DoS Attacks on page 104.

IPv6 ACL Configuration Notes

Consider the following configuration guidelines for IPv6 ACLs:

- IPv4 source guard and IPv6 ACLs are supported together on the same device, as long as they are not configured on the same port or VLAN.
- IPv6 ACLs are supported on tagged ports.
- IPv6 ingress and egress ACLs can be bound to an interface or VLAN without **ipv6 enable** being configured on the physical or virtual interface.
- IPv6 ACLs cannot be used with GRE.
- IPv6 ACLs cannot be employed to implement a user-based ACL scheme.
- On interfaces that have IPv6 ACLs applied on outbound packets, the following features are not supported:
  - ACL mirroring
  - ACL logging
  - Traffic policies
  - Internal priority marking
- It is not possible to configure conflicting ACL filters. When configuring an ACL filter, if the filter parameters match with an existing filter but the action does not match, the following error message is displayed: Error: ACL operation failed for ACL ipv6-test1 since following conflicting filter entry already exists. Please use explicit sequence number to override this error.

Creating and Applying an IPv6 ACL

Before an IPv6 ACL can be applied to an interface or a VLAN, it must first be created.

Complete the following steps to create and apply an IPv6 ACL.

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 access-list** command followed by a name to create the ACL.

   An ACL name must begin with an alphabetical character and must contain no more than 47 characters.

   ```
   device(config)# ipv6 access-list ipv6_test
   ```

3. For each rule, enter the **deny** or **permit** command, specifying the needed parameters. As an option, you can specify the sequence number followed by a permit or deny statement. Otherwise, the sequence numbers will be added automatically in increments of 10 in the order you enter the statements.

   ```
   device(config-ipv6-access-list ipv6_test)# deny tcp host 2001:DB8:e0bb::2 any eq telnet
   device(config-ipv6-access-list ipv6_test)# permit ipv6 any any
   ```

4. Access an interface on which you need to apply the ACL.

   ```
   device(config-ipv6-access-list ipv6_test)# exit
   device(config)# interface ethernet 1/1/1
   ```

5. If needed, enable IPv6 on that interface.

   ```
   device(config-if-e1000-1/1/1)# ipv6 enable
   ```

6. Apply the ACL to the interface.

   ```
   device(config-if-e1000-1/1/1)# ipv6 access-group ipv6_test in
   ```

The following example creates an IPv6 ACL named netw, with remarks preceding each rule.

```
device# configure terminal
device(config)# ipv6 access-list netw

device(config-ipv6-access-list netw)# remark Permits ICMP traffic from 2001:DB8:e0bb::x to 2001:DB8::x.
device(config-ipv6-access-list netw)# permit icmp 2001:DB8:e0bb::/64 2001:DB8::/64

device(config-ipv6-access-list netw)# remark Denies traffic from 2001:DB8:e0ac::2 to 2001:DB8:e0aa:0::24.
device(config-ipv6-access-list netw)# deny ipv6 host 2001:DB8:e0ac::2 host 2001:DB8:e0aa:0::24

device(config-ipv6-access-list netw)# remark Denies all UDP traffic.
device(config-ipv6-access-list netw)# deny udp any any

device(config-ipv6-access-list netw)# remark Permits traffic not explicitly denied by the previous rules.
device(config-ipv6-access-list netw)# permit ipv6 any any
```

The following example applies the IPv6 ACL named netw to incoming traffic on ports 1/1/2 and 1/4/3.

```
device# configure terminal
device(config)# interface ethernet 1/1/2
device(config-if-e1000-1/1/2)# ipv6 access-group netw in
device(config-if-e1000-1/1/2)# exit
device(config)# interface ethernet 1/4/3
device(config-if-e1000-1/4/3)# ipv6 access-group netw in
```

The following example creates an IPv6 ACL named rtr, with remarks preceding each rule.

```
device# configure terminal
device(config)# ipv6 access-list rtr

device(config-ipv6-access-list rtr)# remark Denies TCP traffic from 2001:DB8:21::x to 2001:DB8:22::x.
device(config-ipv6-access-list rtr)# deny tcp 2001:DB8:21::/24 2001:DB8:22::/24

device(config-ipv6-access-list rtr)# remark Denies UDP traffic from ports 5 & 6 to 2001:DB8:22::/24.
device(config-ipv6-access-list rtr)# deny udp any range 5 6 2001:DB8:22::/24

device(config-ipv6-access-list rtr)# remark Permits traffic not explicitly denied by the previous rules.
device(config-ipv6-access-list rtr)# permit ipv6 any any
```

The following example applies the IPv6 ACL named rtr to incoming traffic on ports 1/2/1 and 1/2/2.

```
device# configure terminal
device(config)# interface ethernet 1/2/1
device(config-if-e1000-1/2/1)# ipv6 access-group rtr in
device(config-if-e1000-1/2/1)# exit
device(config)# int eth 1/2/2
device(config-if-e1000-1/2/2)# ipv6 access-group rtr in
```

The following examples show the information displayed for the IPv6 ACL named rtr in **show running-config ipv6** and **show ipv6 access-list rtr** command output.

```
device# show running-config ipv6
!
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 any
!

device# show ipv6 access-list rtr
ipv6 access-list rtr: 3 entries
10: deny tcp 2001:DB8:21::/24 2001:DB8:22::/24
20: deny udp any range rje 6 2001:DB8:22::/24
30: permit ipv6 any any
```

Neighbor Discovery (ND)-Packet DoS Attacks

Checking for neighbor discovery (ND) packets with a hop limit less than 255 helps protect against Denial of Service (DoS) attacks.

Spurious neighbor discovery (ND) packets are among the tactics employed in Denial of Service (DoS) attacks. Based on the assumption that the only valid packet hop-limit value is 255, you can configure IPv6 ACLs to drop such spurious ND packets. The following types of ND packets are checked:

- Neighbor advertisement (NA)
- Neighbor solicitation (NS)
- Router advertisement (RA)
- Router solicitation (RS)

  NOTE
  ND-packet hop-limit check is enabled by default.

IPv6 ACLs have the following concluding implicit rules, which impact on the hop-limit check:

- **permit icmp any any nd-na**: Allows ICMP neighbor discovery acknowledgements.
- **permit icmp any any nd-ns**: Allows ICMP neighbor discovery solicitations.

Hop-limit check enablement varies with the type of ND packet, as indicated in the following table.

TABLE 14 Effect of ND Hop Limits on ND Rules

Packet Type	Implicit or Configured Rules	Hop-Limit Check
ND acknowledgement	implicit **permit**	Yes
ND acknowledgement	**permit icmp ... nd-na**	Yes
ND solicitation	implicit **permit**	Yes
ND solicitation	**permit icmp ... nd-ns**	Yes
router advertisement	**permit icmp ... router-advertisement**	Yes
router advertisement	If not specifically permitted, denied by the implicit **deny ipv6 any any**.	No
router solicitation	**permit icmp ... router-solicitation**	Yes
router solicitation	If not specifically permitted, denied by the implicit **deny ipv6 any any**.	No

Hop-limit check is not applicable to any other types of rules. For example, even if hop-limit check is enabled for the ACL, it does not apply to the two following rules:

```
device(config-ipv6-access-list nd_acl)# permit icmp any any
device(config-ipv6-access-list nd_acl)# permit ipv6 any any
```

Displaying IPv6 ACLs

Use the following commands to display information about IPv6 ACLs:

- **show ipv6 access-lists brief**
- **show ipv6 access-lists**
- **show access-list accounting**
- **show running-config access-list**

To display the IPv6 ACLs configured on a device, enter the **show ipv6 access-lists brief** or **show ipv6 access-lists** command.

```
device# show ipv6 access-lists brief

ipv6 access-list acl3: 100 entries

ipv6 access-list b244: 1 entry

ipv6 access-list cpu-ipv6: 6 entries
enable accounting

ipv6 access-list iv6: 1 entry
enable accounting

device# show ipv6 access-lists
ipv6 access-list v6-ACL1: 1 entries
deny ipv6 any any
ipv6 access-list v6-ACL2: 1 entries
permit ipv6 any any
```

```
ipv6 access-list v6-ACL3: 2 entries
deny ipv6 2001:DB8:10::/64 any
permit ipv6 any any
ipv6 access-list v6-ACL4: 2 entries
deny ipv6 2001:DB8::/64 any
permit ipv6 any any
ipv6 access-list rate-ACL: 1 entries
permit ipv6 any any traffic-policy rate800M
ipv6 access-list v6-ACL5: 8 entries
permit tcp 2001:DB8::/64 any
permit ipv6 2001:DB8::/64 any
permit ipv6 2001:DB8:101::/64 any
permit ipv6 2001:DB8:10::/64 2001:DB8:102::/64
permit ipv6 host 2001:DB8:aa:10::102 host 2001:DB8:101::102
permit ipv6 host 2001:DB8:10::101 host 2001:DB8:101::101 dscp-matching 0
dscp-marking 63 dscp-cos-mapping
permit ipv6 any any dscp-matching 63 dscp-cos-mapping
permit ipv6 any any fragments
```

To display a specific IPv6 ACL configured on a device, enter the **show ipv6 access-lists** command followed by the ACL name. The following example shows the ACL named rtr.

```
device# show ipv6 access-lists 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
```

The following example displays a summary of IPv6 ACLs that are active on VLAN 333.

```
device(config-vlan-333)# show ipv6 access-lists brief

ipv6 access-list acl3: 100 entries

ipv6 access-list b244: 1 entry

ipv6 access-list cpu-ipv6: 6 entries
enable accounting

ipv6 access-list iv6: 1 entry
enable accounting

ipv6 access-list mm: 203 entries

ipv6 access-list mm-cpu: 2 entries

ipv6 access-list scale: 0 entry

ipv6 access-list scale1: 250 entries
enable accounting

ipv6 access-list tp-cpu-ipv6: 1 entry

ipv6 access-list tp-cpu-ipv6-p: 1 entry
```

The following example displays IPv6 ACL filter statements from the running configuration.

```
device(config-vlan-333)# show running-config access-list ipv6
ipv6 access-list acl3
sequence 10 permit ipv6 host 1234::1 any
sequence 20 permit ipv6 host 2234::1 any
sequence 30 permit ipv6 host 1234::2 any
sequence 40 permit ipv6 host 2234::2 any
sequence 50 permit ipv6 host 1234::3 any
sequence 60 permit ipv6 host 2234::3 any
sequence 70 permit ipv6 host 1234::4 any
sequence 80 permit ipv6 host 2234::4 any
sequence 90 permit ipv6 host 1234::5 any
sequence 100 permit ipv6 host 2234::5 any
```

Applying an ACL to a LAG Interface

You can apply an IPv4, IPv6, or MAC ACL to a LAG in LAG interface configuration sub-mode as shown in the following examples. When an ACL is applied to a LAG interface, it filters the traffic on all the member ports in the LAG.

```
device# configure terminal
device(config)# lag 10
device(config-lag-10)# ipv6 access-group tcp_acl
device(config-lag-10)# exit
device(config)#
```

```
device# configure terminal
device(config)# lag 46
device(config-lag-46)# mac access-group maco2 in
device(config-lag-46)# exit
device(config)#
```

Applying ACLs to VLANs

You can apply IPv4, IPv6, and MAC ACLs to VLANs.

IPv4, IPv6, and MAC ACLs can be bound to the same VLAN.

In VLAN configuration sub-mode, you can apply an ACL to all ports in the VLAN, to LAG ports, or to selected ports or LAGs in the VLAN.

Perform the following steps to bind a previously created ACL to a VLAN.

1. Enter global configuration mode

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

2. Enter VLAN configuration sub-mode for the VLAN where the ACL is to be applied.

```
device(config)# vlan 200
device(config-vlan-200)#
```

3. Bind an ACL to the VLAN.
   - Use the **ip access-group** command followed by the ACL name or ID and the direction to bind an IPv4 ACL to the VLAN.

     ```
     device(config-vlan-200)# ip access-group 99 in
     ```

   - Use the **ipv6 access-group** command followed by the ACL name or ID and the direction to bind an IPv6 ACL to the VLAN.

     ```
     device(config-vlan-200)# ipv6 access-group v6 out
     ```

   - Use the **mac access-group** command followed by the ACL name and the direction to bind a MAC ACL to the VLAN.

     ```
     device(config-vlan-200)# mac access-group macl1 in
     ```

The following example binds several ACLs, including IPv6, IPv4, and MAC ACLs, to VLAN 555.

```
device# configure terminal
device(config)# vlan 555 by port
device(config-vlan-555)# lag 10
device(config-vlan-555)# interface ve 555
device(config-vlan-555)# ipv6 access-group scale25 in
device(config-vlan-555)# ipv6 access-group scale15 out
device(config-vlan-555)# mac access-group mac_acl1 in
device(config-vlan-555)# ip access-group 123 in
device(config-vlan-555)# ip access-group 134 out
device(config-vlan-555)# exit
device(config)#
```

The following example applies IPv4, IPv6, and MAC ACLs to LAG interface 10 within the VLAN and enables logging of traffic that matches statements that contain the **log** keyword within the applied ACLs.

```
device# configure terminal
device(config)# vlan 558 by port
device(config-vlan-558)# lag 10
device(config-vlan-558)# ipv6 access-group scale12 in lag 10 logging enable
device(config-vlan-558)# mac access-group mac_acl in lag 10
device(config-vlan-558)# ip access-group 134 in lag 10 logging enable
```

ACL Logging

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

If an ACL rule matches a packet, the software generates a syslog entry and an SNMP trap and starts a five-minute timer. The timer keeps track of all packets that match the ACL entries. After five minutes, the software generates a syslog entry reporting the matches.

Note the following details:

- If the packet rate is high—exceeding the CPU processing rate—the packet count may be inaccurate.
- If there are no matches within the five-minute timer interval, the timer stops, restarting with the next match.
- The timer for logging packets denied by MAC ACLs is a different timer from the ACL logging timer.

Configuration Notes for ACL Logging

Before configuring ACL logging on your device, consider the following configuration notes for IPv4, IPv6, and MAC ACLs.

- ACL logging is a CPU-intensive feature intended for debugging purposes. RUCKUS recommends that you disable ACL logging after the debug session is over.
- To maintain maximum performance, log only specific filters.

- When ACL logging is enabled, packets sent to the CPU are automatically rate-limited to prevent CPU overload.
- You can enable ACL logging on physical interfaces and VLANs.
- If you remove an egress ACL with rules enabled for logging, although new packets are not logged, previous logs will be in the syslog when the sample timer expires.
- The log count may not be accurate for broadcast traffic or for unknown unicast traffic.
- On RUCKUS ICX 7550, RUCKUS ICX 7650, and RUCKUS ICX 7850 devices, ACL logging is not supported for ACLs applied to outbound traffic.
- In a rule that includes one or more of the following parameters, the **log** keyword is ignored:
  - **dscp-matching**
  - **dscp-marking**
  - **802.1p-priority-matching**
  - **802.1p-priority-marking**
  - **802.1p-and-internal-marking**
- ACL logging is not supported for dynamic ACLs with MAC authentication or 802.1X authentication enabled.
- ACL logging is supported for ACLs that are applied to network management access features such as Telnet, SSH, and SNMP.

Enabling ACL Logging

For an ACL to generate a syslog entry, match statements to be logged must contain the **log** keyword, and logging must be enabled for the ACL when it is bound to an interface, LAG, VLAN, or selective ports of a VLAN with the keywords **logging enable** included in the binding command.

Perform the following steps to configure ACL logging.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter configuration sub-mode for a new or existing ACL.
   - To configure an IPv4 ACL, use the appropriate **ip access-list** command followed by the name or ID.

```
device(config)# ip access-list extended v4track
device(config-ipacl-v4track)#
```

   - To configure an IPv6 ACL, use the **ipv6 access-list** command followed by the name or ID.

```
device(config)# ipv6 access-list v6track
device(config-ipv6-access-list v6track)#
```

   - To configure a MAC ACL, use the **mac access-list** command followed by the name.

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

3. Create or update ACL filter statements with the **log** keyword to indicate which matches should be included in the syslog.

   The following example configures logging for an IPv4 extended ACL permit action.

   ```
   device(config)# ip access-list extended v4track
   device(config-ext-ipacl-vtrack)# permit host 10.157.22.26 any log
   ```

   The following example configures logging for an IPv6 ACL permit action.

   ```
   device(config)# ipv6 access-list v6track
   device(config-ipv6-access-list v6track)# permit ipv6 host 10.158.10.10 any log
   ```

   The following example configures logging for a MAC ACL deny action.

   ```
   device(config)# mac access-list mactrack
   device(config-macl-mactrack)# deny any 0000.0000.0088 0000.0000.1111 log
   device(config-macl-mactrack)# permit any any
   ```

4. Enable logging for filtered packets on a specific port, a set of ports, a LAG, or a VLAN.

   The following example enables logging on ports 1/1/1 and 1/1/3 for the MAC ACL mactrack.

   ```
   device(config-macl-maclog)#interface ethernet 1/1/1
   device(config-if-e1000-1/1/1)# mac access-group mactrack in logging enable
   device(config-if-e1000-1/1/1)# interface ethernet 1/1/3
   device(config-if-e1000-1/1/3)# mac access-group mactrack in logging enable
   device(config-if-e1000-1/1/3)# end
   device#
   ```

   The following example binds an IPv6 ACL to a LAG.

   ```
   device# configure terminal
   device(config)# interface lag 46
   device(config-lag-if-lg46)# ipv6 access-group v6track in logging enable
   device(config-lag-if-lg46)# exit
   device(config)#
   ```

   The following example binds an IPv4 ACL to a VLAN.

   ```
   device# configure terminal
   device(config)# vlan 222 by port
   device(config-vlan-222)# vlan 222 by port
   device(config-vlan-222)# interface ve 222
   device(config-vlan-222)# ip access-group v4track in logging enable
   device(config-vlan-222)# exit
   ```

   The following example binds acl1 to incoming traffic on selective ports in VLAN 10 (LAG 1 ports and Ethernet port 1/1/1) and enables logging for the ACL on the same interfaces.

   ```
   ICX(config)# vlan 10
   ICX(config-vlan-10)# tagged ethernet 1/1/1 ethernet 1/1/10 lag 1 lag 10
   ICX(config-vlan-10)# ip access-group acl1 in ethernet 1/1/1 logging enable
   ICX(config-vlan-10)# ip access-group acl1 in lag 1 logging enable
   ICX(config-vlan-10)# exit
   ICX(config)#
   ```

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.

NOTE

The terms *ACL statistics* and *ACL accounting* are used interchangeably in this guide.

For the configuration procedures for ACL statistics, refer to "Traffic Policies" in the *RUCKUS FastIron Traffic Management Configuration Guide*.

ACL Accounting

ACL accounting helps to collect usage information for access control lists configured on the ICX 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 ACLs and provides accounting information for inbound and outbound ACLs.

NOTE
IPv4 and IPv6 ACL accounting are supported in Policy-based Routing (PBR).

Limitations for ACL Accounting

- Traffic Policer and ACL accounting cannot coexist.
- ACL accounting is not supported on dynamic ACLs.
- ACL accounting is not supported on management interfaces.
- ACL accounting is not supported with IP Source Guard (IPSG).

Changing the Accounting Period

By default, the ACL accounting timer collects statistics for 5000 milliseconds (ms) (5 seconds) before generating a report. You can set the timer to any value from 5000 ms through 60000 ms, in increments of 1000 ms.

To set the timer, in global configuration mode, enter the **access-list accounting timer** command followed by the desired duration in milliseconds as shown in the following example.

```
device# configure terminal
device(config)# access-list accounting timer 60000
```

The example sets the timer to collect and report accounting statistics every minute.

Configuring ACL Accounting

If you enable accounting for an ACL, it applies to all rules in that ACL, including implicit rules.

ACL accounting is disabled by default. It is supported and can be enabled on IPv4, IPv6, and MAC ACLs.

1. To enable ACL accounting for a configured ACL, use the **enable accounting** command in IPv4, IPv6, or MAC ACL configuration sub-mode.

   The following example enables accounting for an IPv4 ACL.

   ```
   device(config)# ip access-list standard acl10
   device(config-std-ipacl-acl10)# enable accounting
   ```

   The following example enables accounting for an IPv6 ACL.

   ```
   device(config)# ipv6 access-list acl_1pv6
   device(config-ipv6-access-list acl_ipv6)# enable accounting
   ```

   The following example enables accounting for a MAC ACL.

   ```
   device# configure terminal
   device(config)# mac access-list mac123
   device(config-macl-mac123)# enable accounting
   ```

2. Display the ACL accounting information using the **show access-list accounting** command.

   The accounting statistics are collected every five seconds and synchronized to remote units every one minute.

   The following example displays a brief accounting summary for ACLs applied inbound to VLAN 333.

```
device# show access-list accounting vlan 333 in brief
ACL Accounting Table
=================================================================================================================
=================================================
UnitID: 1 ACL Name: acl_l2            HitCnt: 0   ByteCnt: 0
UnitID: 1 ACL Name: acl_ipv4          HitCnt: 0   ByteCnt: 0
UnitID: 1 ACL Name: acl_ipv6          HitCnt: 0   ByteCnt: 0
UnitID: 2 ACL Name: acl_l2            HitCnt: 0   ByteCnt: 0
UnitID: 2 ACL Name: acl_ipv4          HitCnt: 0   ByteCnt: 0
UnitID: 2 ACL Name: acl_ipv6          HitCnt: 0   ByteCnt: 0
UnitID: 3 ACL Name: acl_l2            HitCnt: 0   ByteCnt: 0
UnitID: 3 ACL Name: acl_ipv4          HitCnt: 0   ByteCnt: 0
UnitID: 3 ACL Name: acl_ipv6          HitCnt: 0   ByteCnt: 0
```

   The following example displays an accounting summary for inbound IPv4 ACLs active on VLAN 333.

```
device# show access-list accounting vlan 333 in ipv4 brief
ACL Accounting Table
=================================================================================================================
=================================================
UnitID: 1 ACL Name: mirror_acl_ipv4     HitCnt: 0     ByteCnt: 0
UnitID: 2 ACL Name: mirror_acl_ipv4     HitCnt: 0     ByteCnt: 0
UnitID: 3 ACL Name: mirror_acl_ipv4     HitCnt: 0     ByteCnt: 0
```

   The following example displays summary inbound MAC ACL accounting statistics for VLAN 224.

```
device# show access-list accounting vlan 224 in  mac brief
ACL Accounting Table
=================================================================================================================
UnitID: 1      ACL Name: mirror_acl_l2        HitCnt: 0          ByteCnt: 0
UnitID: 1      ACL Name: mac                  HitCnt: 1450       ByteCnt: 147144
UnitID: 2      ACL Name: mirror_acl_l2        HitCnt: 0          ByteCnt: 0
UnitID: 2      ACL Name: mac                  HitCnt: 4581       ByteCnt: 456606
UnitID: 4      ACL Name: mirror_acl_l2        HitCnt: 0          ByteCnt: 0
UnitID: 4      ACL Name: mac                  HitCnt: 0          ByteCnt: 0
```

   The following example displays detailed inbound MAC ACL accounting statistics for VLAN 224.

```
device# show access-list accounting vlan 224 in  mac detail
ACL Accounting Table
=================================================================================================
ACL Name: mac
UnitID: 1  Region: 0  Filter Seq Num: 10     Filter Def: deny any 0000.0000.0088 0000.0000.1111 log
   HitCnt: 9    ByteCnt: 774
UnitID: 1  Region: 0  Filter Seq Num: 20     Filter Def: permit any any log
   HitCnt: 1441 ByteCnt: 146370
UnitID: 1  Region: 0  Filter Seq Num: 65001  Filter Def: deny any any
   HitCnt: 0    ByteCnt: 0
UnitID: 2  Region: 0  Filter Seq Num: 10     Filter Def: deny any 0000.0000.0088 0000.0000.1111 log
   HitCnt: 2    ByteCnt: 172
UnitID: 2  Region: 0  Filter Seq Num: 20     Filter Def: permit any any log
   HitCnt: 4579 ByteCnt: 456434
UnitID: 2  Region: 0  Filter Seq Num: 65001  Filter Def: deny any any
   HitCnt: 0    ByteCnt: 0
```

3. To clear ACL accounting statistics for configured ACLs, choose one of the following options:
   - For ACLs configured on a specific interface, enter the **clear access-list accounting** command and the direction in global configuration mode.
   - For all ACLs configured in the device, use the **clear access-list accounting all** command in global configuration mode.

   The following example clears accounting statistics for inbound ACLs on interface 1/1/5.

   ```
   device(config)# clear access-list accounting ethernet 1/1/5 in
   ```

   The following example clears accounting statistics for inbound ACLs on LAG 3.

   ```
   device(config)# clear access-list accounting lag 3 in
   ```

   The following example clears accounting statistics for inbound ACLs on VLAN 22.

   ```
   device(config)# clear access-list accounting vlan 22 in
   ```

   The following example clears all ACL accounting statistics.

   ```
   device(config)# clear access-list accounting all
   ```

The following example enables ACL accounting for a standard IPv4 ACL and applies the ACL to an interface.

```
device# configure terminal
device(config)# ip access-list standard myacl
device(config-std-ipacl-myacl)# permit 10.10.10.0/24
device(config-std-ipacl-myacl)# deny 20.20.20.0/24
device(config-std-ipacl-myacl)# enable accounting
device(config-std-ipacl-myacl)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# ip access-group myacl in
```

The following example enables accounting for an IPv6 ACL and applies the ACL to an interface.

```
device# configure terminal
device(config)# ipv6 access-list acl_ipv6
device(config-ipv6-access-list acl_ipv6)# enable accounting
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# ipv6 access-group acl_ipv6 in
```

The following example enables accounting for a MAC ACL and applies the ACL to a LAG interface.

```
device# configure terminal
device(config)# mac access-list macdata
device(config-macl-macdata)# deny any 0000.0000.0088 0000.0000.1111
device(config-macl-macdata)# permit any any
device(config-macl-macdata)# enable accounting
device(config-macl-macdata)# interface lag 46
device(config-lag-if-lg46)#)# mac access-group macdata in
```

Adding a Comment for an Entry in an ACL

ACL comment text describes entries in an ACL and appears in the output of **show** commands that display ACL information.

To add a comment, enter the **remark** command followed by up to 255 alphanumeric characters on the line immediately preceding an ACL entry as shown in the following example.

```
device(config)# ip access-list extended acl100
device(config-ext-ipacl-acl100)# remark The following line permits TCP packets
device(config-ext-ipacl-acl100)# permit tcp 192.168.4.40/24 2.2.2.2/24
device(config-ext-ipacl-acl100)# remark The following permits UDP packets
device(config-ext-ipacl-acl100)# permit udp 192.168.2.52/24 2.2.2.2/24
device(config-ext-ipacl-acl100)# remark The following line denies all IPv4 traffic
device(config-ext-ipacl-acl100)# deny ip any any
```

The following example shows the **remark** command and text for the ACL named rtr and the resulting **show running-config** output.

```
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# show running-config
ipv6 access-list rtr
 remark This entry permits ipv6 packets from 2001:DB8::2 to any destination
 permit ipv6 host 2001:DB8::2 any
 remark This entry denies udp packets from any source to any destination
 deny udp any any
 remark This entry denies IPv6 packets from any source to any destination
 deny ipv6 any any
```

Deleting a Comment from an ACL Entry

A comment can be deleted only when the filter statement following it is also removed. In some cases, this may mean you must re-enter the filter statement after removing the comment.

To remove a comment and the filter statement it describes, use the **no remark** command in front of the comment, and **no permit** or **no deny** in front of the related filter statement that follows the comment line as shown in the following example.

```
device(config)# ip access-list extended acl100
device(config-ext-ipacl-acl100)# no remark The following line permits TCP packets
device(config-ext-ipacl-acl100)# no permit tcp 192.168.4.40/24 2.2.2.2/24
```

Viewing Comments in an ACL

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

- **show running-config access-list ip**
- **show running-config access-list ipv6**
- **show running-config access-list mac**
- **show access-lists all**
- **show ip access-lists**
- **show ipv6 access-lists**
- **show mac access-lists**

The following example shows the comment text for the ACL acl100 in a **show running-config access-list ip** display.

```
device# show running-config access-list ip
ip access-list extended acl100
remark The following line permits TCP packets
sequence 10 permit tcp 192.168.4.0 0.0.0.255 2.2.2.0 0.0.0.255
remark The following permits UDP packets
sequence 20 permit udp 192.168.2.0 0.0.0.255 2.2.2.0 0.0.0.255
remark The following line denies all IPv4 traffic
sequence 30 deny ip any any
```

The following example shows the comment text for the ACL acl100 in a **show access-lists all** display.

```
device# show access-lists all

Extended IP access list acl100: 3 entries
```

```
ACL Remark: The following line permits TCP packets
10: permit tcp 192.168.4.0 0.0.0.255 2.2.2.0 0.0.0.255
ACL Remark: The following permits UDP packets
20: permit udp 192.168.2.0 0.0.0.255 2.2.2.0 0.0.0.255
ACL Remark: The following line denies all IPv4 traffic
30: deny ip any any
```

Sequence-based ACL Editing

You can specify and modify the sequence in which ACL rules are applied to traffic.

By default, the order in which ACL rules run is determined by the sequence in which they are added to the ACL. Sequence numbers are automatically added to existing ACL rules in the following manner:

- The first rule within each ACL is numbered 10.
- The sequence number for each succeeding rule is incremented by 10.

In new ACLs that you create, specifying rule sequence numbers is optional. However, sequence numbers are assigned automatically in the previously mentioned order.

Sequence numbers have the following advantages:

- If you need a new rule to run between existing rules, you assign the new rule a sequence number between those two rules.
- New rules are implemented seamlessly, with no need to re-apply ACLs.
- If you delete a rule, there is no need to re-apply the ACL.

If you add a rule to an ACL without specifying its sequence, the rule is added at the end of the list. Such a rule is automatically assigned the next multiple of 10 as a sequence number.

You can regenerate sequence numbers within a specified ACL. Although by default, such regeneration assigns 10 to the first rule and increments each succeeding rule by 10, you can specify both the initial sequence number and the sequence gap parameter. Sequence regeneration settings (first sequence number and sequence gap number) are persistent, even following reload of the device.

Inserting Rules into ACLs

To insert rules into an ACL, follow these steps.

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

   ```
   device# configure terminal
   ```

2. Enter ACL configuration sub-mode for the IPv4, IPv6, or MAC ACL you want to modify.

   ```
   device(config)# ip access-list standard ip_acl_1
   device(config)# ipv6 access-list ipv6_acl_1
   device(config)# mac access-list macdata
   ```

3. Enter the appropriate **show** command to display the rules of the relevant ACL.
   - Enter the **show ip access-lists** command followed by the name of an IPv4 ACL.
   - Enter the **show ipv6 access-lists** command followed by the name of an IPv6 ACL.
   - Enter the **show mac access-lists** command followed by the name of a MAC ACL.

```
device(config-std-ipacl-ip_acl_1)# show ip access-lists ip_acl_1
Standard IP access list ip_acl_1: 3 entries
10: permit 1.1.1.0  0.0.0.255
11: permit host 2.2.2.1
20: deny   2.2.2.0  0.0.0.255

device(config-ipv6-access-list ipv6_acl_1)# show ipv6 access-lists ipv6_acl_1
ipv6 access list ipv6_acl_1: 3 entries
10: permit ipv6 2001:DB8::/64 any
11: permit ipv6 2001:DB8:101::/64 any
20: permit ipv6 host 2001:DB8:aa:10::102 host 2001:DB8:101::102

device(config-macl-mac123)# show mac access-lists mac123
mac access-list mac123: 1 entries
permit 1111.2222.3333 ffff.ffff.ffff 4444.5555.6666 ffff.ffff.ffff
```

4. To insert a rule between two adjacent rules, complete the following steps:
   a) In ACL configuration sub-mode, enter the **regenerate-seq-number** command.

```
device(config-ipv6-access-list ipv6_acl_1)# regenerate-seq-number
```

   b) Re-enter the appropriate **show** command to display the updated contents of the ACL.

```
device(config-ipv6-access-list ipv6_acl_1)# show ipv6 access-lists ipv6_acl_1
10: permit ipv6 2001:DB8::/64 any
20: permit ipv6 2001:DB8:101::/64 any
30: permit ipv6 host 2001:DB8:aa:10::102 host 2001:DB8:101::102
```

5. Create the new rule, beginning with the keyword **sequence** and the needed sequence number.

```
device(config-ipv6-access-list ipv6_acl_1)# sequence 25 permit ipv6 2001:DB8:10::/64
2001:DB8:102::/64
```

Deleting Rules from ACLs

To delete rules from an IPv4, IPv6, or MAC ACL, complete the following steps.

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

```
device# configure terminal
```

2. Enter ACL configuration sub-mode for the IPv4, IPv6, or MAC ACL you want to modify.

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

device(config)# ipv6 access-list ipv6_acl_1

device(config)# mac access-list macdata
```

3. If needed, enter the appropriate **show** command to display the rules of the relevant ACL.
   - Enter the **show ip access-lists** command followed by the name of an IPv4 ACL.
   - Enter the **show ipv6 access-lists** command followed by the name of an IPv6 ACL.
   - Enter the **show mac access-lists** command followed by the name of a MAC ACL.

   ```
   device(config-std-ipacl-ip_acl_1)# show ip access-lists ip_acl_1
   Standard IP access list ip_acl_1: 3 entries
   10: permit 1.1.1.0  0.0.0.255
   11: permit host 2.2.2.1
   20: deny   2.2.2.0  0.0.0.255
   ```

4. Remove the rule in one of the following ways:
   - To delete a rule by sequence number, enter the **no sequence** command, specifying the sequence number.

     ```
     device(config-std-ipacl-ip_acl_1)# no sequence 11
     ```

   - To delete a rule by content, enter the **no permit** command, specifying the rule content.

     ```
     device(config-std-ipacl-ip_acl_1)# no permit host 2.2.2.1
     ```

5. If you want to clean up the sequence numbers, for example, so that you can re-use a sequence number that you have deleted, enter the **regenerate-seq-number** command.

   ```
   device(config-std-ipacl-ip_acl_1)# regenerate-seq-number
   ```

Displaying TCAM Information for ACLs

You can use different forms of the **show access-list tcam** command to display information on the following items:

- IPv4 ACLs
- IPv6 ACLs
- MAC ACLs
- ACLs applied to the CPU
- ACLs applied to an interface or LAG
- ACLs applied to a stack unit
- ACLs applied to incoming traffic
- ACLs applied to outbound traffic
- Statistics on ACL rules stored in TCAM

Refer to the following examples for more information.

The following example provides TCAM information for ACL 136. The **show access-list tcam acl-name** command shows which ports have the ACL programmed in TCAM, the type of ACL, which direction the ACL is applied, and how many rules, including the default rule, are programmed in TCAM for the ACL.

```
device(config-vlan-222)# show access-list tcam acl-name 136
Ingress
UnitId Feature   SRule ERule Filters Contiguous RefCnt Bind If
------ --------  ----- ----- ------- ---------- ------ -------
1      UACL-IPv4 1123  2125  1003    YES        1      e 1/1/18
2      UACL-IPv4 1123  2125  1003    YES        1      e 2/1/18
```

The following example provides detailed information for the same ACL. The detailed information includes ACL rules and associated ACL sequence numbers and ports.

```
device(config-vlan-222)# show access-list tcam acl-name 136 detail
Ingress:
UnitId Region Feature   Filter ID Rule  RefCnt Bind If
------ ------ -------   --------- ----- ------ -------
1      0      UACL-IPv4  8        1123  1      e 1/1/18
1      0      UACL-IPv4 10        1124  1      e 1/1/18
1      0      UACL-IPv4 20        1125  1      e 1/1/18
1      0      UACL-IPv4 30        1126  1      e 1/1/18
1      0      UACL-IPv4 40        1127  1      e 1/1/18
1      0      UACL-IPv4 50        1128  1      e 1/1/18
1      0      UACL-IPv4 60        1129  1      e 1/1/18
1      0      UACL-IPv4 70        1130  1      e 1/1/18
1      0      UACL-IPv4 80        1131  1      e 1/1/18
```

The following example displays information on TCAM rules for the IPv6 ACL named vlan333-ipv6.

```
device(config-vlan-333)# show running-config vlan 333
vlan 333 by port
 tagged ethe 1/1/10 ethe 2/1/10 ethe 3/1/10 lag 25
 interface ve 333
 ipv6 access-group vlan333-ipv6 in  <-- Applied VLAN shown in output
 ip access-group vlan333-ipv4 in
 ip access-group frag deny
!
!

device(config-vlan-333)# show access-list tcam acl-name vlan333-ipv6
Ingress:
UnitId Feature   SRule ERule Filters Contiguous RefCnt Bind If
------ -------   ----- ----- ------- ---------- ------ -------
1      UACL-IPv6 641   786   146     YES        1      e 1/1/10
2      UACL-IPv6 1012  1157  146     YES        2      e 2/1/10 e 2/1/38
3      UACL-IPv6 1147  1292  146     YES        2      e 3/1/10 e 3/1/40
device(config-vlan-333)#
```

The following example shows detailed TCAM information for the same ACL.

```
device(config-vlan-333)# show access-list tcam acl-name vlan333-ipv6 detail
Ingress:
UnitId Region Feature         Filter ID Rule  RefCnt Bind If
------ ------ -------         --------- ----- ------ -------
1      0      UACL-IPv6       1         641   1      e 1/1/10
1      0      UACL-IPv6       2         642   1      e 1/1/10
1      0      UACL-IPv6       3         643   1      e 1/1/10
1      0      UACL-IPv6       6         644   1      e 1/1/10
1      0      UACL-IPv6       12        645   1      e 1/1/10
1      0      UACL-IPv6       13        646   1      e 1/1/10
1      0      UACL-IPv6       14        647   1      e 1/1/10
```

The following example breaks out TCAM rule information for a MAC ACL found in the running configuration for VLAN 333.

```
device(config-vlan-333)# show running-config vlan 333
vlan 333 by port
 tagged ethe 1/1/10 ethe 2/1/10 ethe 3/1/10 lag 25
 interface ve 333
 ipv6 access-group vlan333-ipv6 in
 mac access-group mac_acl in
 ip access-group vlan333-ipv4 in
 ip access-group frag deny
!
!

device(config-vlan-333)# show access-list tcam acl-name mac_acl
Ingress:
UnitId Feature         SRule ERule Filters Contiguous RefCnt Bind
If
------ -------         ----- ----- ------- ---------- ------
```

```
-------
1      UACL-MAC        1119  1129  11      YES        1      e
1/1/10
2      UACL-MAC        1490  1500  11      YES        2      e 2/1/10 e
2/1/38
3      UACL-MAC        1625  1635  11      YES        2      e 3/1/10 e
3/1/40
device(config-vlan-333)#

device(config-vlan-333)# show access-list tcam acl-name mac_acl detail
Ingress:
UnitId Region Feature         Filter ID Rule  RefCnt Bind If
------ ------ -------         --------- ----- ------ -------
1      0      UACL-MAC        10        1119  1      e 1/1/10
1      0      UACL-MAC        20        1120  1      e 1/1/10
1      0      UACL-MAC        30        1121  1      e 1/1/10
1      0      UACL-MAC        40        1122  1      e 1/1/10
1      0      UACL-MAC        50        1123  1      e 1/1/10
1      0      UACL-MAC        60        1124  1      e 1/1/10
1      0      UACL-MAC        70        1125  1      e 1/1/10
1      0      UACL-MAC        80        1126  1      e 1/1/10
1      0      UACL-MAC        90        1127  1      e 1/1/10
1      0      UACL-MAC        100       1128  1      e 1/1/10
1      0      UACL-MAC        65001     1129  1      e 1/1/10
2      0      UACL-MAC        10        1490  2      e 2/1/10 e 2/1/38                              ^C
device(config-vlan-333)#
```

The following example displays TCAM usage for a specified stack unit. Information includes available TCAM space for IPv4, IPv6, and MAC ACLs. For a dual-PP device, the command output includes information on the region (0 or 1).

```
device# show access-list tcam usage unit 4
UnitId Region Group Id Direction  Type             :            Allocated Total Free
------ ------ -------- ---------  ----             :            --------- ----- ----
4      0      1        Pre-Ingres L2_IPv4 FIlters  :            1         512   511
4      0      2        Pre-Ingres VCAP_MISC        :            0        1024  1024
4      0      3        Ingress    IPv4 Filters     :            9        2816  2807 <--
4      0      4        Ingress    IPv6 Filters     :            0        1408  1408 <--
4      0      5        Ingress    L2 Filters       :           30        2816  2786 <--
4      0      6        Ingress    ICAP All Combo   :           51         768   717
4      0      7        Ingress    IPSec Filters    :            0        1408  1408
4      0      8        Egress     IPv4 Filters     :            0         256   256 <--
4      0      9        Egress     IPv6 Filters     :            0         256   256 <--
4      0     10        Egress     L2 Filters       :            3         256   253 <--
4      1      1        Pre-Ingres L2_IPv4 FIlters  :            1         512   511
4      1      2        Pre-Ingres VCAP_MISC        :            0        1024  1024
4      1      3        Ingress    IPv4 Filters     :         1031        2816  1785
4      1      4        Ingress    IPv6 Filters     :            7         896   889
4      1      6        Ingress    ICAP All Combo   :           51         512   461
4      1      7        Ingress    IPSec Filters    :            0         896   896
4      1      8        Egress     IPv4 Filters     :            4         256   252
4      1      9        Egress     IPv6 Filters     :          247         256     9
4      1     10        Egress     L2 Filters       :            3         256   253
device#
```

The following example displays TCAM information for a specified interface. Use this command to verify ACLs applied on an interface and how many filters are programmed in TCAM for each ACL.

```
device# show access-list tcam interface ethernet 4/1/10
Ingress:
UnitId AclName      Feature    SRule ERule  Filters Contiguous Merged Acl
------ -------      -------    ----- -----  ------- ---------- ---------
4      STK_ZTP_0403 ZTP         36    36    1       YES
4      STK_IPC_0401 STK_HIGIG    5     5    1       YES
4      123          UACL-IPv4   84   104   21       YES
4      mac_acl      UACL-MAC   105   115   11       YES

Egress:
UnitId AclName      Feature    SRule ERule Filters Contiguous Merged Acl
------ -------      -------    ----- ----- ------- ---------- ---------
```

```
4      140          UACL-IPv4  128   129    2      YES
4      egress       UACL-IPv6  118   127   10      YES
```

The following example displays more detailed information for the same interface, including all rules and filters (by sequence number) for each ACL bound to the interface.

```
device# show access-list tcam interface ethernet 4/1/10 detail
Ingress:
UnitId Region AclName        Feature     Filter Id Rule
------ ------ -------        -------     --------- -----
4      1      STK_ZTP_0403   ZTP         1         36
4      1      STK_IPC_0401   STK_HIGIG   1          5
4      1      123            UACL-IPv4  10         84
4      1      123            UACL-IPv4  20         85
4      1      123            UACL-IPv4  30         86
4      1      123            UACL-IPv4  40         87
4      1      123            UACL-IPv4  50         88
```

The following example displays TCAM information for a specified LAG interface.

```
device# show access-list tcam interface lag 8060
Ingress:
UnitId AclName     Feature       SRule  ERule Filters Contiguous Merged Acl
------ -------     -------       -----  ----- ------- ---------- ---------
2      qos_dscp_34 QOS-DSCP/PCP   909    909  1       YES
3      qos_dscp_34 QOS-DSCP/PCP   907    907  1       YES

Egress:
UnitId AclName     Feature       SRule  ERule Filters Contiguous Merged Acl
------ -------     -------       -----  ----- ------- ---------- ---------
2      125         UACL-IPv4      1587   1822 236     YES
2      egress      UACL-IPv6      1823   2026 204     YES
3      125         UACL-IPv4      1585   1820 236     YES
3      egress      UACL-IPv6      1821   2024 204     YES
device#
```

The following example displays detailed information for a specified LAG.

```
device# show access-list tcam interface lag 8060 detail
Ingress:
UnitId Region AclName     Feature      Filter Id Rule
------ ------ -------     -------      --------- -----
2      0      qos_dscp_34 QOS-DSCP/PCP 10        909
3      0      qos_dscp_34 QOS-DSCP/PCP 10        907

Egress:
UnitId Region AclName     Feature      Filter Id Rule
------ ------ -------     -------      --------- -----
2      0      125         UACL-IPv4      2       1587
2      0      125         UACL-IPv4    110       1588
2      0      125         UACL-IPv4    120       1589
```

The following example displays a list of all ACLs and associated rules, including default rules, programmed on unit 1 in an inbound direction.

```
device# show access-list tcam ingress unit 1

Ingress:
UnitId AclName                        Feature      SRule ERule    Filters Contiguous RefCnt Bind If
------ -------                        -------      ----- -----    ------- ---------- ------ -------
1 SFLOW_RULE                          SFLOW        34       34       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 INGRESS_CPU_RULE                    CPU_RULES     5        5       1    YES        33     e 1/1/1 to
1/1/24 e 1/2/1 to 1/2/8
1 MANAGEMENT                          UACL-IPv4  3165     3356     192    YES         1     e 1/1/2
1 SYS_MGMT_VLAN                       VLAN          6        6       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SYSTEM-L3-UDP-BC                    UDP_BC       35       35       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SPX_ZTP_0402                        SPX_IPC_MAC  36       36       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
```

```
1 STK_ZTP_0403                          ZTP          37       37       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 STK_IPC_0401                          STK_HIGIG     7        7       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 MCAST_ACL_RULES_IGMP                  IGMP         38       38       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 MCAST_ACL_RULES_PIM_V4                PIMV4        39       39       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 MCAST_ACL_RULES_RES_MC_V4             RES_MC_V4    40       40       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 MCAST_ACL_RULES_SC_SP_MLD_V1          MLD          41       41       2    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 MCAST_ACL_RULES_SC_SP_MLD_V2          MLD          44       44       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 MCAST_ACL_RULES_SC_SP_PIM_V6          PIMV6        45       45       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SYS_CPU_VLAN_BPDU                     VLAN         46       46       1    YES         1
1 SYS_PVST                              XSTP         47       47       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SYS_MRP                               MRP           8        8       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SYS_UDLD                              UDLD          9        9       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SYS_EOAM_LOOPBACK                     EOAM         48       48       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SYS_SMAC_SUP                          FDB          10       10       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SYSTEM-L3-IPV6-RES-MC                 IPV6_RES_MC  49       49       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SYSTEM-L3-IRDP                        IRDP          1        1       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SYSTEM-L3-ARP-PRIORITY                ARP          11       11       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 FLEXAUTH_802.1X_BPDU_RULE_UNIT_1      FLEXAUTH     50       50       1    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SYS_PROTO_REPRIO                      L2_PROTO     51       52       2    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 L2MCAST-ACL-RULES-SPATHA-SICA-UMC-V6 MC_UMC        53       55       3    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 SYSTEM-L3-ND                          ND           56       58       3    YES        30     e 1/1/1 to
1/1/24 e 1/2/2 e 1/2/4 to 1/2/8
1 scale22                               UACL-IPv6  2357     3160     804    YES         2     e 1/1/11 e
1/2/2
1 131                                   UACL-IPv4  3161     3164       4    YES         2     e 1/1/11 e
1/2/2
```

The following example displays TCAM information for ACLs applied in an outbound direction on unit 1. The command output shows all ACLs programmed in TCAM for the specified unit in the specified direction, including system default rules.

```
device# show access-list tcam egress unit 1
Egress:
UnitId AclName           Feature    SRule ERule Filters Contiguous RefCnt Bind If
------ -------           -------    ----- ----- ------- ---------- ------ -------
1      ECPU_PORTID_RULE  CPU_RULES   84    85   2       YES        1
1      ECPU_CLASSID_RULE CPU_RULES   86    86   1       YES        1
device#
```

The **show access-list tcam rule-statistics** command is used to display hardware-level accounting statistics. The output is displayed for a specific rule in a specific region on a specific unit.

```
device# show access-list tcam rule-statistics 3161 unit 1 region 0
Rule: 3161 Stat: 0
device#
```

The **show access-list tcam rule** command displays detailed output for each rule programmed in TCAM. The command is local to each unit. The following example displays information on rules for region 0 of unit 1.

```
device# show access-list tcam rule 3161 unit 1 region 0
EID 0x00000c59: gid=0x3,
```

```
slice=0, slice_idx=0xc9, part =0 prio=0x1fe0216, flags=0x210602, Installed, Enabled
tcam: color_indep=1,
StageIngress
InPorts
DATA=0x00000000000000000000000000000000000000000000000000008000000000800
MASK=0x00000000000000000000000000000000000000000000000003fe000001ffffff
Stage
IpType
Offset0: 325 Width0: 4
DATA=0x00000000
MASK=0x0000000e
InterfaceClassL2
Offset0: 32 Width0: 12
DATA=0x0000000e
MASK=0x00000fff
action={act=CosQCpuNew, param0=31(0x1f), param1=0(0x00), param2=0(0x00), param3=0(0x00)}
action={act=SwitchToCpuCancel, param0=0(0x00), param1=0(0x00), param2=0(0x00), param3=0(0x00)}
action={act=DynamicHgTrunkCancel, param0=0(0x00), param1=0(0x00), param2=0(0x00), param3=0(0x00)}
action={act=Drop, param0=0(0x00), param1=0(0x00), param2=0(0x00), param3=0(0x00)}
policer=
statistics={stat id 3079 slice = 6 idx=0 entries=1}{Packets}{Bytes}
device#
```

The following example displays TCAM information for the ACL cpu-ipv4 applied to incoming traffic on the CPU of the active controller for the stack.

```
device# configure terminal
device(config)# interface cpu active
device(config-if-cpu-active)# ip access-group cpu-ipv4 in

device(config-if-cpu-active)# show access-list tcam acl-name cpu-ipv4
Egress:
UnitId Feature   SRule ERule Filters Contiguous RefCnt Bind If
------ --------   ----- ----- ------- ---------- ------ -------
1      UACL-IPv4 2218  2220  3       YES        1
2      UACL-IPv4 1250  1252  3       YES        1
```

MAC ACLs

- Layer 2 ACL Overview 125
- MAC Scaling by ICX Device 125
- MAC ACL Default Action 126
- MAC ACL Configuration Notes and Limitations 126
- Configuring and Applying MAC ACLs 126
- Displaying MAC ACL Information 128

Layer 2 ACL Overview

NOTE
MAC ACLs replaced separate MAC filter statements in FastIron release 08.0.95. Refer to the *RUCKUS FastIron Software Upgrade Guide* for additional information on the changes.

MAC ACLs are implemented much like IP ACLs using the following steps:

1. Create the MAC ACL using the **mac access-list** command.
2. Define permit and deny rules using the **deny** and **permit** commands.
3. Apply the MAC ACL to one or more interfaces or VLANs using the **mac access-group** command.

MAC ACLs are supported on all ICX device physical interface types and LAGs.

MAC ACLs are named ACLs. Names must begin with an alphabetical character and contain no more than 47 characters.

MAC ACLs can be applied to individual interfaces, sets of interfaces, LAGs, VLANs, and selected ports within a VLAN and can be combined with IPv4 and IPv6 ACLs.

NOTE
For information on Layer 3 filtering as well as on configuring ACLs for LAGs and VLANs, enabling logging, enabling accounting, modifying ACL statements, and displaying information about TCAM usage, refer to ACLs on page 85.

NOTE
For the use of ACLs under Flexible Authentication, refer to Dynamic ACLs in Authentication on page 196.

MAC Scaling by ICX Device

All ICX devices support 256 MAC match statements per ACL and 3,072 filter statements per device.

Hardware scaling allows for the following number of filter statements per device:

- ICX 7550: 2,048
- ICX 7650: 1,536
- ICX 7850: 1,536
- ICX 8200: 481

NOTE
By default on all platforms, 30 entries are reserved in the Layer 2 ingress group.

MAC ACL Default Action

The default action when no ACLs are configured on a device is to permit all traffic. Once you apply a MAC ACL to a port or VLAN, the device drops all Ethernet traffic on the port or VLAN that does not match a MAC permit filter. Given these defaults, follow these guidelines for configuring and applying ACLs:

- If you want to tightly control access, configure MAC ACLs consisting of permit entries for the access you want to permit. The MAC ACLs implicitly deny all other access.
- If you want to secure access in environments with many users, you may want to configure MAC ACLs that consist of explicit deny entries and then add an entry to permit all access (for example, "**permit any any**") to the end of each MAC ACL so that the software permits all MAC addresses that do not match a previous deny statement.

MAC ACL Configuration Notes and Limitations

Keep the following points in mind when configuring MAC ACLs:

- MAC ACL filtering on RUCKUS ICX devices is performed in hardware.
- On RUCKUS ICX devices, MAC ACLs can match on source and destination MAC addresses and Ethertype.
- MAC ACLs 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 any" statement is included in a MAC ACL and applied.
- MAC ACLs cannot be applied on the out-of-band management port.
- If you apply a MAC ACL to a port or VLAN that already has a MAC ACL applied, the older MAC ACL is replaced by the new ACL.
- MAC ACLs 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 ACL rule that allows the routing protocol MAC and the neighbor system MAC address.
- MAC ACLs are supported on tagged ports.
- MAC ACLs do not support filtering based on Layer 4 information.
- MAC ACL logging does not work in the following case: Ingress logging is enabled for IPv4, IPv6, and MAC ACLs for the same VLAN, interface, or selective port in a VLAN, and transmitted IPv4 or IPv6 traffic matches IPv4 or IPv6 ACL rules.

Configuring and Applying MAC ACLs

Complete the following steps to configure and apply a MAC ACL.

1. Enter global configuration mode.

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

2. Enter the **mac access-list** command followed by a unique ACL name to enter MAC ACL configuration sub-mode. Define the set of MAC ACL filter statements to permit or deny traffic based on source and/or destination MAC address and optional Ethertype.

   In the source or destination MAC address field, you can specify a particular MAC address with an exact mask or a range of MAC addresses with a variable mask, or you can specify **any** to match any MAC address. Specify the mask using hexidecimal numbers 0 through f. For example, to match on the first two bytes of the address 0010.0075.3676, use the mask ffff.0000.0000. In this case, the filter matches on all MAC addresses that contain "0010" as the first two bytes. The filter accepts any value for the remaining bytes of the MAC address.

   ```
   device(config)# mac access-list mac01
   device(config-macl-mac01)# deny any 0010.0075.3676 ffff.0000.0000
   device(config-macl-mac01)# deny any 0000.0023.fbcd ffff.ffff.ffff
   device(config-macl-mac01)# deny 0000.0123.0121 ffff.ffff.fff5 any
   device(config-macl-mac01)# deny any 0180.c200.0000 ffff.ffff.ffff
   device(config-macl-mac01)# deny any 0000.0034.5678 ffff.ffff.ffff
   device(config-macl-mac01)# deny any 0000.0045.6789 ffff.ffff.ffff
   device(config-macl-mac01)# permit any any
   ```

   The previous example contains five deny statements and permits traffic from any other source or destination address. The first deny statement denies all source MAC addresses that begin with 0010. The third deny statement denies traffic to any destination address when the source address matches values in the range 0000.0123.0126 through 0000.0123.012f (values 0120 through 0125 are allowed in the last byte). The remaining deny statements deny any source MAC address sent to a specific MAC address.

3. To apply a previously created MAC ACL, enter interface configuration sub-mode and enter the **mac access-group** command followed by the name of the MAC ACL and the **in** keyword. This applies the MAC ACL to all inbound traffic.

   ```
   device(config)# interface ethernet 1/1/1
   device(config-if-e1000-1/1/1)# mac access-group mac01 in
   ```

   The previous example applies the MAC ACL created in this task to inbound traffic on port 1/1/1.

   When a MAC ACL is applied to or removed from an interface, a syslog message is generated.

   ```
   SYSLOG: <14> May  7 16:22:03 ACL: acl1 applied to eth 1/1/1 by un-authenticated user from console
   session.

   SYSLOG: <14> May  7 16:22:43 ACL: acl1 removed from eth 1/1/1 by un-authenticated user from console
   session.
   ```

   The syslog messages indicate that a MAC ACL was applied to the specified port and then removed by an unauthenticated user during the specified session type. The session type can be Console, Telnet, SSH, Web, or SNMP, among others.

   NOTE
   A MAC ACL can be applied to a port, LAG, VLAN, or selected ports of a VLAN.

The following example denies Layer 2 traffic if the conditions of the MAC ACL deny statements are matched and permits traffic from all other source MAC addresses as per the last match statement. The MAC ACL is applied to inbound traffic on port 1/1/2.

```
device# configure terminal
device(config)# mac access-list mac02
device(config-macl-mac02)# deny 0010.0075.3676 ffff.0000.0000
device(config-macl-mac02)# deny any 0000.0023.fbcd ffff.ffff.ffff
device(config-macl-mac02)# deny any 0180.c200.0000 ffff.ffff.fff0
device(config-macl-mac02)# deny any 0000.0034.5678 ffff.ffff.ffff
device(config-macl-mac02)# deny any 0000.0045.6789 ffff.ffff.ffff
device(config-macl-mac02)# permit any any
device(config-macl-mac02)# interface ethernet 1/1/2
device(config-if-e1000-1/1/2)# mac access-group mac02 in
```

The following example defines an ACL that allows all traffic from a specific Ethertype, in this case, 0800, or IPv4.

```
device# configure terminal
device(config)# mac access-list mac21
device(config-macl-mac21)# permit any any ether-type 0800
```

NOTE

Valid Ethertype values for MAC ACLs are in the range 600 through ffff.

The following example creates MAC ACL mac02, which denies traffic from a specific MAC address to another specific MAC address and permits all other traffic. It applies the MAC ACL to inbound traffic on LAG 46.

```
device# configure terminal
device(config)# mac access-list mac02
device(config-macl-mac02)# deny any 0000.0000.0088 0000.0000.1111
device(config-macl-mac02)# permit any any
device(config-macl-mac02)# interface lag 46
device(config-lag-if-lg46)# mac access-group mac02 in
device(config-lag-if-lg46)# exit
device(config)#
```

The following example applies a previously created MAC ACL to inbound traffic in a VLAN.

```
device# configure terminal
device(config)# vlan 555 by port
device(config-vlan-555)# tagged ethernet 1/2/2 lag 10
device(config-vlan-555)# interface ve 555
device(config-vlan-555)# mac access-group mac02 in
device(config-vlan-555)# exit
device(config)#
```

The following example applies the same MAC ACL to inbound traffic on selected ports (port 1/1/21 and LAG 5 ports) in VLAN 41.

```
device# configure terminal
device(config)# vlan 41 by port
device(config-vlan-41)# tagged ethernet 1/1/21 lag 5
device(config-vlan-41)# mac access-group mac02 in ethernet 1/1/21 lag 5
device(config-vlan-41)# exit
device(config)#
```

Displaying MAC ACL Information

Use the following commands to display information about MAC ACLs:

- **show mac access-lists**
- **show running-config access-list**

NOTE

Refer to Displaying TCAM Information for ACLs on page 118 for information on MAC ACL TCAM usage.

Use the **show mac access-lists brief** command for a summary of information about all configured MAC ACLs.

```
device# show mac access-lists brief

 Total ACL count= 2
 Total ACL filter count= 10

 ACL Name                          Filters  Bindings(In)   Bindings(Out)  Acc

 ================================  =======  =============  =============  ===
 mac_acl                           10       0              0              dis
 vlan22                            0        1              0              dis
```

Use the **show mac access-lists** command to display information for all configured MAC ACLs.

```
device# show mac access-lists
mac access-list mac123: 1 entries
permit 1111.2222.3333 ffff.ffff.ffff 4444.5555.6666 ffff.ffff.ffff
mac access-list mac456: 1 entries
permit 1234.5678.9000 ffff.ffff.ffff any
```

Use the **show mac access-lists** command followed by a MAC ACL name to display information for a specific MAC ACL.

```
device# show mac access-lists mac456
mac access-list mac456: 1 entries
permit 1234.5678.9000 ffff.ffff.ffff any log mirror
```

Use the **show mac access-lists bindings** command to display information about where specific MAC ACLs are applied. The command can be entered for a particular MAC ACL, LAG, or interface.

```
device(config)# show mac access-lists bindings
ACL Name                          Bind If               Dir
================================  ====================  ===
acl1                              eth 1/1/1             in
acl1                              lag 1                 in
```

The following example displays the filter statements for configured MAC ACLs.

```
device(config-vlan-333)# show running-config access-list mac
!
mac access-list mac_acl
 permit 0000.0000.1111 ffff.ffff.ffff any log
 permit 0000.0000.4444 ffff.ffff.ffff any log
 permit 0000.0000.5555 ffff.ffff.ffff any log
 permit 0000.0000.6666 ffff.ffff.ffff any log
 permit 0000.0000.7777 ffff.ffff.ffff any log
 permit 0000.0000.8888 ffff.ffff.ffff any log
 permit 0000.0000.9999 ffff.ffff.ffff any log
 permit 0000.0009.1010 ffff.ffff.ffff any log
 permit 0000.0009.1011 ffff.ffff.ffff any log
 permit any any log
mac access-list vlan22
!
device(config-vlan-333)#
```

Policy-Based Routing

- Policy-Based Routing Overview 131
- Route Maps 132
- Configuration Guidelines for IPv4 PBR 133
- Configuration Guidelines for IPv6 PBR 138

Policy-Based Routing Overview

Policy-based routing (PBR) is the process of altering a packet's path based on criteria other than the destination address. PBR allows you to use access control lists (ACLs) and route maps to selectively route an IP packet. The ACLs classify the traffic and the route maps that match on the ACLs set routing attributes for the traffic.

A PBR policy specifies the routing attributes for traffic that matches the policy. Using standard ACLs with route maps in PBR, an IP packet is routed based on its source IP address. With extended ACLs, you can route IP packets based on all of the clauses in the extended ACL.

NOTE
For more details about ACLs, refer to the ACLs on page 85 chapter.

The following are the basic steps used to configure a PBR policy:

1. Configure ACLs that specify all the conditions required to match the desired packets to be routed using PBR.
2. Configure a route map that matches on the ACLs and sets the route information.
3. Enable PBR by applying the route map globally or on an untagged interface or virtual interface.

PBR Permit and Deny Actions

To configure PBR, you define the policies using ACLs and route maps, and then enable PBR on individual interfaces. The platform programs the ACLs on the interfaces, and routes traffic that matches the ACLs according to the instructions provided by the “set” statements in the route map entry.

The configuration of a set of match criteria and corresponding routing information (for example, next hops) is referred to as a stanza. You can create multiple stanzas and assign an “Instance_ID” that controls the program positioning within the route map. Furthermore, when the route map is created, you specify a deny or permit construct. In addition, the ACL used for the “match” criteria also contains a deny or permit construct.

The deny or permit nomenclature has a different meaning within the context of the PBR operation than it does within the normal context of user-applied ACLs (where deny and permit are directly correlated to the forwarding actions of forward and drop). The following table lists the behavior between the permit and deny actions specified at the route-map level, in conjunction with the permit and deny actions specified at the ACL rule level.

TABLE 15 Behavior of Permit and Deny Actions in Different Contexts

Route-map Permit and Deny Actions	ACL Clause Permit and Deny Actions	Resulting Ternary Content Addressable Memory (TCAM) Action
Permit	Permit	The “set” statement of the route-map entry is applied.
Permit	Deny	The packet is “passed” and routed normally. The contents of the “set” command are not applied. A rule is programmed in the TCAM as a “permit” with no result actions preventing any further statements of the route-map ACL from being applied.

TABLE 15 Behavior of Permit and Deny Actions in Different Contexts (continued)

Route-map Permit and Deny Actions	ACL Clause Permit and Deny Actions	Resulting Ternary Content Addressable Memory (TCAM) Action
Deny	Permit	The packet is “passed” and routed normally. There should be no “set” commands following the “match” command of a deny route-map. A rule is programmed in the TCAM as a “permit” with no result actions preventing any further statements of the route-map ACL from being applied.
Deny	Deny	No TCAM entry is provisioned; no other route map ACL entries will be compared against. If no subsequent matches are made, the packet is forwarded as normal.

LAG Formation with PBR Policy

PBR can be applied on a LAG virtual interface only if the port is untagged. When a LAG is formed, the PBR policy on the LAG virtual interface applies to all the member ports. If a different PBR policy exists on a member port at the time of a LAG formation, that policy is overridden by the PBR policy on the LAG virtual interface. If the LAG virtual interface does not have a PBR policy, then the member ports will not have a PBR policy.

When a LAG is removed, the PBR policy that was applied to the LAG virtual interface is unbound (removed) from former member ports. If global PBR is configured, the member ports adhere to the global PBR; otherwise, no PBR policy is bound to former member ports. When a LAG is formed, all ports must have the same PBR configuration. During LAG creation, the configuration on the LAG virtual interface is replicated to all ports.

Route Maps

Route maps enable you to define routing policy for the traffic causing a packet to be forwarded to a predetermined next-hop interface, bypassing the path determined by normal routing.

Each entry in a route map statement contains a combination of match and set statements. A route map specifies the match criteria that correspond to ACLs, followed by a set statement that specifies the resulting action if all of the match clauses are met. You can define multiple match and next-hop specifications (set statement) on the same interface. When a PBR policy has multiple next hops to a destination, PBR selects the first live next hop specified in the policy that is up. If none of the policy's direct routes or next hops is available, the packets are forwarded as per the routing table.

Match Statement

The IP standard or extended ACLs can be used to establish the match criteria. Using the standard ACLs with route maps in PBR, an IP packet is routed based on its source IP address. Using the extended ACLs, you can route IP packets based on all of the clauses in the extended ACL.

Set Statement

Traffic that matches a match statement in the route map is forwarded as defined by set commands. Multiple set commands can be configured and when a match condition is met, the device works sequentially through the list of set commands until it finds the first "next hop" that is operational and uses it. If that "next hop" goes down, the next available next-hop as defined in a set command is chosen and if all next hop interfaces in the list are down, the packet is routed as determined in the IP Route Table. If a next-hop interface that was down comes back up, the next hop selection process begins again and restarts its selection process from the top of the list.

The set clauses are evaluated in the following order:

1. Set clauses where the next hop is specified.
2. Set interface NULL0.

The order in which you enter the **set ip next-hop** commands determines the order of preference. If no next hops are reachable, the egress interface is selected based on the IP route table. The set interface NULL0 clause — regardless of which position it was entered — is always placed as the last selection in the list.

NOTE
The "match" and "set" statements described in this chapter are the only route-map statements supported for PBR. Other route-map statements described in the documentation apply only to the protocols with which they are described.

NOTE
If none of the clauses of a PBR route-map definition contains both “match” and “set” statements together, PBR does not work and the packets are forwarded as per the routing table.

The following are the PBR next hops that can be specified in a route map for matched traffic:

- IPv4 address
- IPv6 address
- Null interface
- VRF interface

Configuration Guidelines for IPv4 PBR

- PBR is not supported together with ingress ACLs on the same port.
- Global PBR is not supported with per-port-per-VLAN ACLs.
- Global PBR is supported only on the default VRF.
- A PBR policy on an interface takes precedence over a global PBR policy.
- Although you can apply an IPv4 policy route map to more than 64 interfaces, only the first 64 active route maps (including instances) will be used.
- You cannot apply PBR on a port if that port already has ingress ACLs, ACL-based rate limiting, DSCP-based QoS, or MAC address filtering.
- The number of route maps that you can define is limited by the available system memory, which is determined by the system configuration and how much memory other features use. When a route map is used in a PBR policy, the PBR policy uses up to 21 instances of a route map, up to five ACLs in a matching policy of each route map instance, and up to six next hops in a set policy of each route map instance. The CLI allows you to configure more than 21 route maps instances; however, only 21 instances can be active at one time. An active route map is a route map with at least one valid ACL and one valid next hop. The CLI also allows you to configure more than six next hops in a route map; however, the extra next hops will not be placed in the PBR database. A route map might be used by other features such as BGP or OSPF, which may use more than six next hops.
- Denial of Service (DoS) protection is supported in PBR.
- IPv4 ACL accounting is supported in PBR.
- ACLs with the **log** option configured in the **ip access-list** or **ipv6 access-list** command should not be used for PBR purposes.
- PBR ignores explicit or implicit **deny ip any any** ACL entries to ensure that the traffic is compared to all ACLs for route maps that use multiple ACLs. PBR also ignores any deny clauses in an ACL. Traffic that matches a deny clause is routed normally using Layer 3 paths.
- PBR always selects the first next hop from the next-hop list that is up. If a PBR policy next hop goes down, the policy uses another next hop if available. If no next hops are available, the device routes the traffic in the normal way.
- PBR is not supported for fragmented packets. If the PBR ACL filters on Layer 4 information such as TCP/UDP ports, fragmented packets are routed normally.
- You can change route maps or ACL definitions dynamically and do not need to rebind the PBR policy to an interface.

Configuring an IPv4 PBR Policy with an IPv4 Address as the Next Hop

The following steps configure an IPv4 PBR by setting an IPv4 address as the next hop in the route map. This task uses Access Control Lists (ACLs), which are explained in greater detail in the *RUCKUS FastIron Security Configuration Guide* for your platform.

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

```
device# configure terminal
```

2. Define the required IPv4 ACLs to be added to the route map.

```
device(config)# ip access-list standard 99
device(config-std-ipacl-99)# permit 10.157.23.0 0.0.0.255
device(config-std-ipacl-99)# exit
```

3. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all the match clauses are met.

```
device(config)# route-map test-route permit 99
```

4. Add IPv4 ACLs to match the IP address that is permitted by the ACL.

```
device(config-routemap test-route)# match ip address 99
```

5. Set the IPv4 address of the next hop to which the traffic that matches a match statement in the route map must be routed.

```
device(config-routemap test-route)# set ip next-hop 192.168.3.1
```

NOTE
If the IP address used in this command is the IP address of a configured IPsec or GRE tunnel, the configuration will choose IPsec or GRE tunnel interface 192.168.3.1 as the next-hop address for matching packets. If you want to set the next hop using a GRE tunnel or IPsec tunnel, use the **set next-hop-ip-tunnel** command.

Optionally, the route map can be configured to forward the packet to the neighbor router without decrementing the Time-to-Live (TTL) value in the packet header for the traffic matched by the policy using the **no-ttl-decrement** option.

By default, the TTL value in the packet header is decremented (decreased) for routed traffic and the packet will be discarded when the TTL is exhausted. TTL functions as a hop count limit and every routing hop decrements the TTL value by one. When the TTL value becomes zero, the packet is discarded to prevent routing loops. The **no-ttl-decrement** option in the **set ip next-hop** command disables the TTL decrement and the packets will be forwarded without decrementing TTL for the traffic matched by the policy.

6. Enter the **exit** command to return to global configuration mode.

```
device(config-routemap test-route)# exit
```

7. Enable PBR by applying the route map globally or on an untagged interface or virtual interface.
   - Enable IPv4 PBR globally to apply the route map to all interfaces.

```
device(config)# ip policy route-map test-route
```

   - Enable IPv4 PBR locally by applying the route map on an interface.

```
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ip policy route-map test-route
```

   - Enable IPv4 PBR locally by applying the route map on a virtual interface.

```
device(config)# interface ve 1
device(config-vif-1)# ip policy route-map test-route
```

The following example shows the configuration steps to configure an IPv4 PBR policy by setting an IPv4 address as the next hop in the route map.

```
device# configure terminal
device(config)# ip access-list standard 99
device(config-std-ipacl-99)# permit 10.157.23.0 0.0.0.255
device(config-std-ipacl-99)# exit
device(config)# route-map test-route permit 99
device(config-routemap test-route)# match ip address 99
device(config-routemap test-route)# set ip next-hop 192.168.3.1
device(config-routemap test-route)# exit
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ip policy route-map test-route
device(config-if-e1000-1/1/3)# end
```

The following example shows the configuration steps to configure an IPv4 PBR policy in which the route map is configured to forward the packet without decrementing the Time-to-Live (TTL) value in the packet header.

```
device(config)# ip access-list standard 99
device(config-std-ipacl-99)# permit 10.157.23.0 0.0.0.255
device(config-std-ipacl-99)# exit
device(config)# route-map test-route permit 99
device(config-routemap test-route)# match ip address 99
device(config-routemap test-route)# set ip next-hop 192.168.3.1 no-ttl-decrement
device(config-routemap test-route)# exit
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ip policy route-map test-route
device(config-if-e1000-1/1/3)# end
```

Configuring an IPv4 PBR Policy with the NULL0 Interface as the Next Hop

The following steps configure an IPv4 PBR policy by setting the NULL0 interface as the next hop in the route map.

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

   ```
   device# configure terminal
   ```

2. Define the required IPv4 ACLs to be added to the route map.

   ```
   device(config)# ip access-list standard 99
   device(config-std-ipacl-99)# permit 10.157.23.0 0.0.0.255
   device(config-std-ipacl-99)# exit
   ```

3. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all the match clauses are met.

   ```
   device(config)# route-map test-route permit 99
   ```

4. Add IPv4 ACLs to match the IP address that is permitted by the ACL.

   ```
   device(config-routemap test-route)# match ip address 99
   ```

5. Set the next hop as NULL0 interface to send the traffic to the null interface, thus dropping the packet instead of forwarding it.

   ```
   device(config-routemap test-route)# set interface null0
   ```

6. Enter the **exit** command to return to global configuration mode.

   ```
   device(config-routemap test-route)# exit
   ```

7. Enter configuration mode on the interface where you want to enable PBR by applying the route map.

```
device(config)# interface ethernet 1/1/3
```

   PBR can be enabled globally by which the route map is applied to all interfaces using the **ip policy route-map** command from the global configuration mode.

8. Enable PBR on the interface and specify the route map to be used.

```
device(config-if-e1000-1/1/3)# ip policy route-map test-route
```

The following example shows the configuration steps to configure an IPv4 PBR policy to send all traffic from 10.157.23.0 0.0.0.255 to the NULL0 interface, thus dropping the traffic instead of forwarding it.

```
device# configure terminal
device(config)# ip access-list standard 99
device(config-std-ipacl-99)# permit 10.157.23.0 0.0.0.255
device(config-std-ipacl-99)# exit
device(config)# route-map test-route permit 99
device(config-routemap test-route)# match ip address 99
device(config-routemap test-route)# set set interface null0
device(config-routemap test-route)# exit
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ip policy route-map test-route
device(config-if-e1000-1/1/3)# end
```

Configuring an IPv4 PBR Policy with a Tunnel as the Next Hop

Traffic can be configured to route over a GRE tunnel using PBR. The following steps configure a GRE tunnel interface as the next hop of a PBR route map.

You must configure the GRE tunnel before configuring the traffic to route over a tunnel. For more information about GRE tunnel configuration, refer to the *RUCKUS FastIron Layer 3 Routing Configuration Guide*.

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

```
device# configure terminal
```

2. Define the required IPv4 ACLs to be added to the route map.

```
device(config)# ip access-list standard 99
device(config-std-ipacl-99)# permit 10.157.23.0 0.0.0.255
```

3. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all the match clauses are met.

```
device(config)# route-map test-route permit 99
```

4. Add IPv4 ACLs to match the IP address that is permitted by the ACL.

```
device(config-routemap test-route)# match ip address 99
```

5. Set the configured tunnel as the next hop for a route map.

```
device(config-routemap test-route)# set next-hop-ip-tunnel 1
```

6. Enter the **end** command to return to global configuration mode.

```
device(config-routemap test-route)# end
```

7. Enter configuration mode on the interface where you want to enable IPv4 PBR by applying the route map.

```
device(config)# interface ethernet 1/1/3
```

   IPv4 PBR can be enabled globally by which the route map is applied to all interfaces using the **ip policy route-map** command from global configuration mode.

8. Enable PBR on the interface and specify the route map to be used.

```
device(config-if-e1000-1/1/3)# ip policy route-map test-route
```

The following example shows the configuration steps to route routing traffic over a GRE tunnel using PBR.

```
device(config)# interface tunnel 1
device(config-tnif-1)# vrf forwarding blue
device(config-tnif-1)# tunnel source ethernet 1/1/1
device(config-tnif-1)# tunnel destination 10.2.2.1
device(config-tnif-1)# tunnel mode gre ip
device(config-tnif-1)# ip address 10.4.4.4/24
device(config-tnif-1)# exit
device(config)# ip access-list standard 99
device(config-std-ipacl-99)# permit 10.157.23.0 0.0.0.255
device(config-std-ipacl-99)# exit
device(config)# route-map test-route permit 99
device(config-routemap test-route)# match ip address 99
device(config-routemap test-route)# set next-hop-ip-tunnel 1
device(config-routemap test-route)# end
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# vrf forwarding blue
device(config-if-e1000-1/1/3)# ip policy route-map test-route
device(config-if-e1000-1/1/3)# end
```

Configuring an IPv4 PBR Policy by Setting a VRF-aware Next Hop in a Route Map

The following steps configure a next hop in a VRF for a route map.

A VRF must be configured and an interface must be assigned to the VRF before setting a VRF-aware next hop in a route map.

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

```
device# configure terminal
```

2. Define the required IPv4 ACLs to be added to the route map.

```
device(config)# ip access-list standard 99
device(config-std-ipacl-99)# permit 10.157.23.0 0.0.0.255
device(config-std-ipacl-99)# exit
```

3. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all of the match clauses are met.

```
device(config)# route-map test-route permit 99
```

4. Add IPv4 ACLs to match the IP address that is permitted by the ACL.

```
device(config-routemap test-route)# match ip address 99
```

5. Set the configured VRF as the next hop for a route map.

```
device(config-routemap test-route)# set ip next-hop 192.168.3.1 vrf vrf_c
```

Displaying IPv4 PBR Information

Use the **show policy-routing ipv4** command with available parameters to display information about policy-based routing applied to IPv4 interfaces.

The following examples show the IPv4 policy-based routing applied on device interfaces.

```
device(config)# show policy-routing ipv4 global
        Configured, pbrmap, Active
        Active Port List:
        ethernet
                1/1/2, 1/1/3, 1/1/4, 1/1/5, 1/1/6, 1/1/7, 1/1/8, 1/1/9
        ve
                Ve 131, Ve 132

device# show policy-routing ipv4 interface all
ve 11 :
        Configured, pbrmap, Active

device# show policy-routing ipv4 interface ethernet
ethernet 1/1/1:
        Configured, pbrmap, Active

device# show policy-routing ipv4 interface ethernet 2/2/1
ethernet 2/2/1:
        Configured, pbrmap, Active
```

For more information on the **show policy-routing** command, refer to the *RUCKUS FastIron Command Reference*.

Configuration Guidelines for IPv6 PBR

IPv6 PBR allows you to manually configure how IPv6 packets that match certain criteria can be forwarded instead of following the IPv6 Routing Table Manager (RTM) routes.

The next hop can be configured as an IPv6 address, a tunnel ID, or as a drop interface. IPv6 ACLs can be configured as match criteria for the packets. An IPv6 policy route map can be applied globally or on an interface. If an IPv6 PBR policy is configured on an interface, it will take precedence over the global-level IPv6 PBR policy configuration.

The **ip policy route-map** and **ipv6 policy route-map** commands are independent of each other and can be applied on an interface and globally in any order, individually or simultaneously. Although IPv4 and IPv6 PBR work independently, and route maps can support IPv4 and IPv6 next hops and ACLs, the command used to apply the route map does affect which next hops and ACLs are supported:

- If a route map is applied using the **ip policy route-map** command, only IPv4 next hops and ACLs will be used and IPv6 next-hops and ACLs will be skipped.
- If a route map is applied using the **ipv6 policy route-map** command, only IPv6 next hops and ACLs will be used and IPv4 next hops and ACLs will be skipped.

IPv6 PBR follows the same configuration considerations as IPv4 PBR with the following exceptions:

- Each IPv6 PBR route map instance can support up to six IPv6 ACLs.
- IPv6 PBR is supported for the following criteria:
  - VRF support
  - Accounting
  - Stacking or HA

Configuring an IPv6 PBR Policy with an IPv6 Address as the Next Hop

The following steps configure an IPv6 address as the next hop in the route map in a policy. This task uses Access Control Lists (ACLs), which are explained in greater detail in the "ACLs" chapter.

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

```
device# configure terminal
```

2. Define the required IPv6 ACLs to be added to the route map.

```
device(config)# ipv6 access-list acl8
device(config-ipv6-access-list ac18)# permit ipv6 2001:DB8:12d:1300::/64 any
device(config-ipv6-access-list ac18)# exit
```

3. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all the match clauses are met.

```
device(config)# route-map test-route permit acl8
```

4. Add IPv6 ACLs to match the IP address that is permitted by the ACL.

```
device(config-routemap test-route)# match ipv6 address acl8
```

5. Set the IPv6 address of the next hop to which the traffic that matches a match statement in the route map must be routed.

```
device(config-routemap test-route)# set ipv6 next-hop 2001:DB8:12d:1300:1
```

6. Enter the **exit** command to return to global configuration mode.

```
device(config-routemap test-route)# exit
```

7. Enable PBR by applying the route map globally or on an untagged interface or virtual interface.
   - Enable IPv6 PBR globally to apply the route map to all interfaces.

```
device(config)# ipv6 policy route-map test-route
```

   - Enable IPv6 PBR locally by applying the route map on an interface.

```
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ipv6 policy route-map test-route
```

   - Enable IPv6 PBR locally by applying the route map on a virtual interface.

```
device(config)# interface ve 1
device(config-vif-1)# ipv6 policy route-map test-route
```

The following example shows the configuration steps to configure a PBR policy by setting an IPv6 address as the next hop in the route map.

```
device# configure terminal
device(config)# ipv6 access-list acl8
device(config-ipv6-access-list ac18)# permit ipv6 2001:DB8:12d:1300::/64 any
device(config-ipv6-access-list ac18)# exit
device(config)# route-map test-route permit acl8
device(config-routemap test-route)# match ipv6 address acl8
device(config-routemap test-route)# set ipv6 next-hop 2001:DB8:12d:1300:1
device(config-routemap test-route)# exit
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ipv6 policy route-map test-route
device(config-if-e1000-1/1/3)# end
```

Configuring an IPv6 PBR Policy with the NULL0 Interface as the Next Hop

Perform the following steps to configure an IPv6 PBR to set the NULL0 interface as the next hop in the route map.

The NULL0 interface does not forward traffic or receive traffic. Setting the next hop as the NULL0 interface drops the packet instead of forwarding the packet.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Define the required IPv6 ACLs to be added to the route map.

   ```
   device(config)# ipv6 access-list acl1
   ```

3. Define a permit clause with an IPv6 address.

   ```
   device(config-ipv6-access-list acl1)# permit ipv6 2001:db8:85a3:0:0:8a2e:370:7334 any
   ```

4. Define a deny clause.

   ```
   device(config-ipv6-access-list acl1)# deny ipv6 any any
   ```

5. Return to global configuration mode.

   ```
   device(config-ipv6-access-list acl1)# exit
   ```

6. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all the match clauses are met.

   ```
   device(config)# route-map null-route permit acl1
   ```

7. Add IPv6 ACLs to match the IP address that is permitted by the ACL.

   ```
   device(config-routemap null-route)# match ipv6 address acl1
   ```

8. Set the next hop as the NULL0 interface to send the traffic to the null interface, and this action will drop the packet instead of forwarding it.

   ```
   device(config-routemap null-route)# set interface null0
   ```

9. Enter the **exit** command to return to global configuration mode.

   ```
   device(config-routemap null-route)# exit
   ```

10. Enter configuration mode on the interface where you want to enable PBR by applying the route map.

    ```
    device(config)# interface ethernet 1/1/3
    ```

    IPv6 PBR can be enabled globally by which the route map is applied to all interfaces using the **ip policy route-map** command from global configuration mode.

11. Enable PBR on the interface and specify the route map to be used.

    ```
    device(config-if-e1000-1/1/3)# ipv6 policy route-map null-route
    ```

The following example shows the configuration steps to configure a PBR policy to send all IPv6 traffic from 2001:db8:85a3:0:0:8a2e:370:7334 to the NULL0 interface, thus dropping the traffic instead of forwarding it.

```
device# configure terminal
device(config)# ipv6 access-list acl1
device(config-ipv6-access-list acl1)# permit ipv6 2001:db8:85a3:0:0:8a2e:370:7334 any
device(config-ipv6-access-list acl1)# deny ipv6 any any
device(config-ipv6-access-list acl1)# exit
device(config)# route-map null-route permit acl1
device(config-routemap null-route)# match ip address acl1
device(config-routemap null-route)# set interface null0
device(config-routemap null-route)# exit
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# ipv6 policy route-map null-route
device(config-if-e1000-1/1/3)# end
```

Configuring an IPv6 PBR Policy with a Tunnel as the Next Hop

Traffic can be configured to route IPv6 packets over an IPsec tunnel or GRE tunnel using PBR. The following steps configure an IPsec or GRE tunnel interface as the next hop of a PBR route map.

You must configure the GRE tunnel before configuring the traffic to route over a tunnel. For more information about GRE tunnel configuration, refer to the *RUCKUS FastIron Layer 3 Routing Configuration Guide*.

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

   ```
   device# configure terminal
   ```

2. Define the required IPv6 ACLs to be added to the route map.

   ```
   device(config)# ipv6 access-list acl99
   device(config-ipv6-access-list-acl99)# permit ipv6 2001:DB8:90::22/64 any
   device(config-ipv6-access-list-acl99)# exit
   ```

3. Enter the **route-map** command to define the route and specify the match criteria and the resulting action if all the match clauses are met.

   ```
   device(config# route-map tunnel-route permit acl99
   ```

4. Add IPv6 ACLs to match the IP address that is permitted by the ACL.

   ```
   device(config-routemap tunnel-route)# match ipv6 address acl99
   ```

5. Set the configured tunnel as the next hop for a route map.

   ```
   device(config-routemap tunnel-route)# set next-hop-ipv6-tunnel 1
   ```

6. Enter the **end** command to return to global configuration mode.

   ```
   device(config-routemap tunnel-route)# end
   ```

7. Enter configuration mode on the interface where you want to enable the IPv6 PBR policy by applying the route map.

   ```
   device(config)# interface ethernet 1/1/3
   ```

   The IPv6 PBR policy can be enabled globally when the route map is applied to all interfaces using the **ipv6 policy route-map** command from global configuration mode.

8. Enable PBR on the interface and specify the route map to be used.

   ```
   device(config-if-e1000-1/1/3)# ipv6 policy route-map tunnel-route
   ```

Displaying IPv6 PBR Information

Use the **show policy-routing ipv6** command with available parameters to display information about policy-based routing applied to IPv6 interfaces.

The following examples show the IPv6 policy-based routing applied on device interfaces.

```
device# show policy-routing ipv6 global
        Configured, pbrv6_rmap, Active
        Active Port List:
        ve
                Ve 131, Ve 132

device# show policy-routing ipv6 interface all
ve 11 :
        Configured, pbrv6_rmap, Active

device# show policy-routing ipv6 interface ve
ve 11 :
        Configured, pbrv6_rmap, Active

device# show policy-routing ipv6 interface ve 132
ve 132:
        Configured, pbrv6_rmap, Active
```

For more information on the **show policy-routing command**, refer to the *RUCKUS FastIron Command Reference*.

Media Access Control Security

- MACsec Overview 143
- How MACsec Works 144
- Configuring MACsec 148
- Enabling MACsec and Configuring Group Parameters 149
- Enabling and Configuring Group Interfaces for MACsec 153
- Sample MACsec Configuration 154
- Displaying MACsec Information 154

MACsec Overview

Media Access Control Security (MACsec) is a Layer 2 security technology that provides point-to-point security on Ethernet links between nodes.

NOTE
MACsec is supported on ICX 7550, ICX 7650, and ICX 7850 devices.

MACsec, defined in the IEEE 802.1AE-2006 standard, is based on symmetric cryptographic keys. MACsec Key Agreement (MKA) protocol, defined as part of the IEEE 802.1x-2010 standard, operates at Layer 2 to generate and distribute the cryptographic keys used by the MACsec functionality installed in the hardware.

As a hop-to-hop Layer 2 security feature, MACsec can be combined with Layer 3 security technologies such as IPsec for end-to-end data security.

Supported MACsec Hardware Configurations

MACsec key-enabled security can be deployed on a point-to-point LAN between two connected ICX devices over interfaces that share a preconfigured static key, the Connectivity Association Key (CAK).

On a licensed ICX 7550, ICX 7650, or ICX 7850 device, 10-Gbps ports can be configured for MACsec. Licenses are available per device as described in the *RUCKUS FastIron Software Licensing Guide*.

NOTE

1. On ICX 7550 devices, MACsec is available only on 4 X 10GF modules installed in slot 3.
2. On ICX 7650 devices, MACsec is available only on 10-Gbps fiber ports, that is, ports 25 through 48 of the base module for ICX 7650-48F devices or on slot 2 when a 4 X 10GF module is installed.
3. MACsec is available on 10-Gbps ports of ICX 7850-48FS devices only.

MACsec RFCs and Standards

FastIron MACsec complies with the following industry standards:

- IEEE Std 802.1X-2010: Port-Based Network Access Control
- IEEE Std 802.1AE-2006: Media Access Control (MAC) Security
- RFC 3394: Advanced Encryption Standard (AES) Key Wrap Algorithm
- RFC 5649: Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm

MACsec Considerations

Review the following considerations before deploying MACsec:

- As a prerequisite, MACsec must be licensed on each device where it is used.
- MACsec introduces an additional transit delay, due to the increase in the MAC Service Data Unit (MSDU) size.
- MACsec and Flexible authentication cannot be configured on the same port.
- On an ICX 7550 device, ports on a 4 X 10GF removable module installed in slot 3 can be used for MACsec or stacking but not both simultaneously.

How MACsec Works

MACsec capabilities prevent Layer 2 security threats, such as passive wiretapping, denial of service, intrusion, man-in-the-middle, and playback attacks.

MACsec protects communications using several configurable techniques. Data origin is authenticated and data is transported over secured channels. Frames are validated as MACsec Ethernet frames. The integrity of frame content is verified on receipt. Frame sequence is monitored using an independent replay protection counter. Invalid frames are discarded or monitored.

Data traffic carried within the MACsec frame is encrypted and decrypted using an industry-standard cipher suite.

How MACsec Handles Data and Control Traffic

All traffic is controlled on an active MACsec port; that is, data is encrypted, or its integrity is protected, or both. If a MACsec session cannot be secured, all data and control traffic is dropped.

When MACsec is active on a port, the port blocks the flow of data traffic. Data traffic is not forwarded by the port until a MACsec session is secured. If an ongoing session is torn down, traffic on the port is again blocked until a new secure session is established.

Control traffic (such as STP, LACP, or UDLD traffic) is not transmitted by an active MACsec port until a MACsec session is secured. While a session is being established, only 802.1X protocol packets are transmitted from the port. Once a secure session is established, control traffic flows normally through the port.

MACsec Key Agreement Protocol

MACsec Key Agreement (MKA) protocol installed on a device relies on an IEEE 802.1X Extensible Authentication Protocol (EAP) framework to establish communication.

MACsec peers on the same LAN belong to a unique connectivity association. Members of the same connectivity association identify themselves with a shared Connectivity Association Key (CAK) and Connectivity Association Key Name (CKN). The CAK is a static key that is preconfigured on each MACsec-enabled interface. MACsec authentication is based on mutual possession and acknowledgment of the preconfigured CAK and Connectivity Association Key Name (CKN).

Each peer device establishes a single unidirectional secure channel for transmitting MACsec frames (Ethernet frames with MACsec headers that usually carry encrypted data) to its peers within the connectivity association. A connectivity association consists of two secure channels, one for inbound traffic, and one for outbound traffic. All peers within the connectivity association use the same cipher suite, either Galois/Counter Mode Advanced Encryption Standard 128 or 256 (GCM-AES-128 or GCM-AES-256), for MACsec-authenticated security functions.

MACsec Key Agreement (MKA) protocol uses the Connectivity Association Key to derive transient session keys called Secure Association Keys (SAKs). SAKs and other MKA parameters are required to sustain communication over the secure channel and to perform encryption and other MACsec

security functions. SAKs, along with other essential control information, are distributed in MKA protocol control packets, also referred to as MKPDUs.

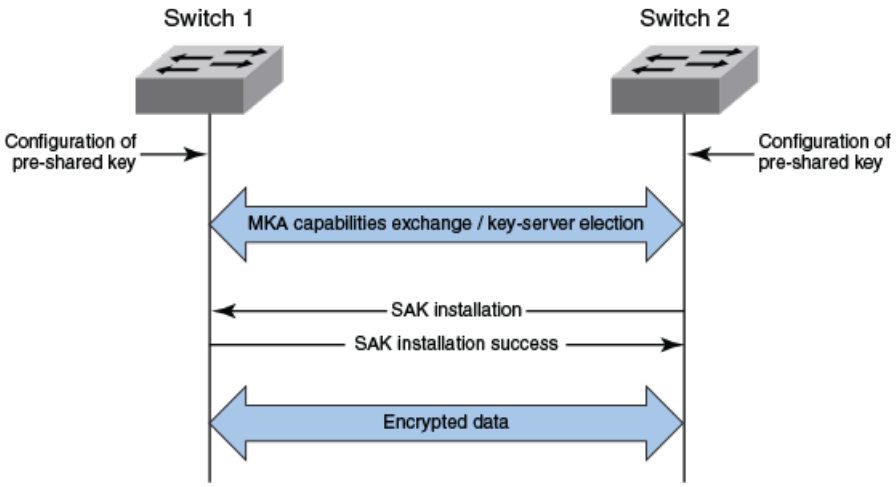
MKA Message Exchange between Two Switches

When two MACsec peers confirm possession of a shared CAK and CKN, MKA protocol initiates key-server election.

The key-server is responsible for determining whether MACsec encryption is used and what cipher suite is used to encrypt data. The key-server is also responsible for generating Secure Association Keys (SAKs) and distributing them to the connected device. Once a SAK is successfully installed, the two devices can exchange secure data.

The following figure shows the message flow between two switches during MACsec communication.

FIGURE 1 MKA Pre-shared Key and Key Name Exchange between Two Switches



Secure Channels

Communication on each secure channel takes place as a series of transient sessions called secure associations. These sessions can only be established with a unique Secure Association Key (SAK) assigned to the session.

Secure associations expire and must be re-established after transmission of a certain number of frames or after a peer disconnects and reconnects.

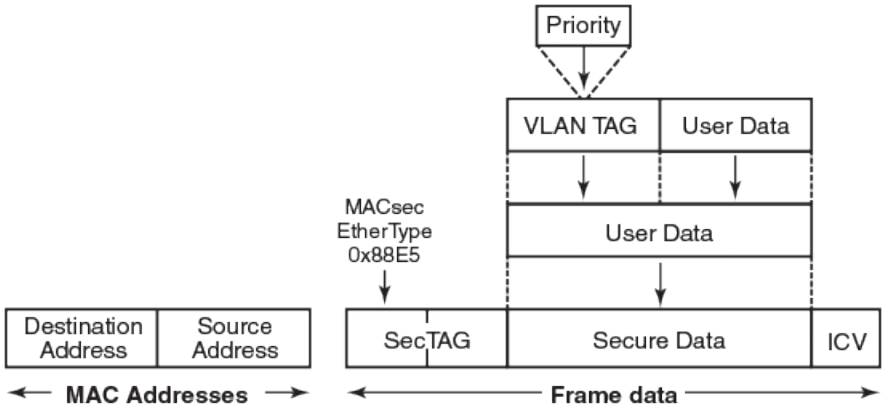
The secure association is designated by a Secure Association Identifier (SAI), formed from the Secure Channel Identifier (SCI) combined with an Association Number (AN). When a MACsec frame is received by a peer interface, the device identifies the session key from the SAI carried in the MACsec frame and uses the key to decrypt and authenticate the received frame.

MACsec Frame Format

When MACsec is enabled, FastIron hardware transforms each Ethernet frame by adding a security tag (secTAG) to the frame.

The following figure shows how the Ethernet frame is converted into a MACsec frame.

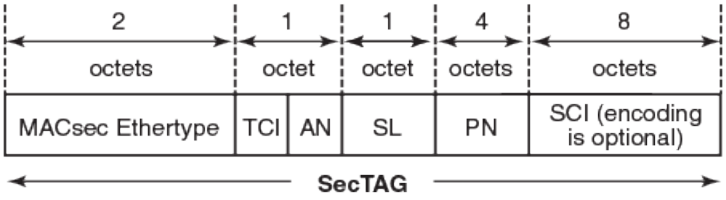
FIGURE 2 MACsec Frame Format



The security tag passes MACsec-related information to MACsec peers.

The following figure defines the fields in a security tag.

FIGURE 3 MACsec Security Tag Format

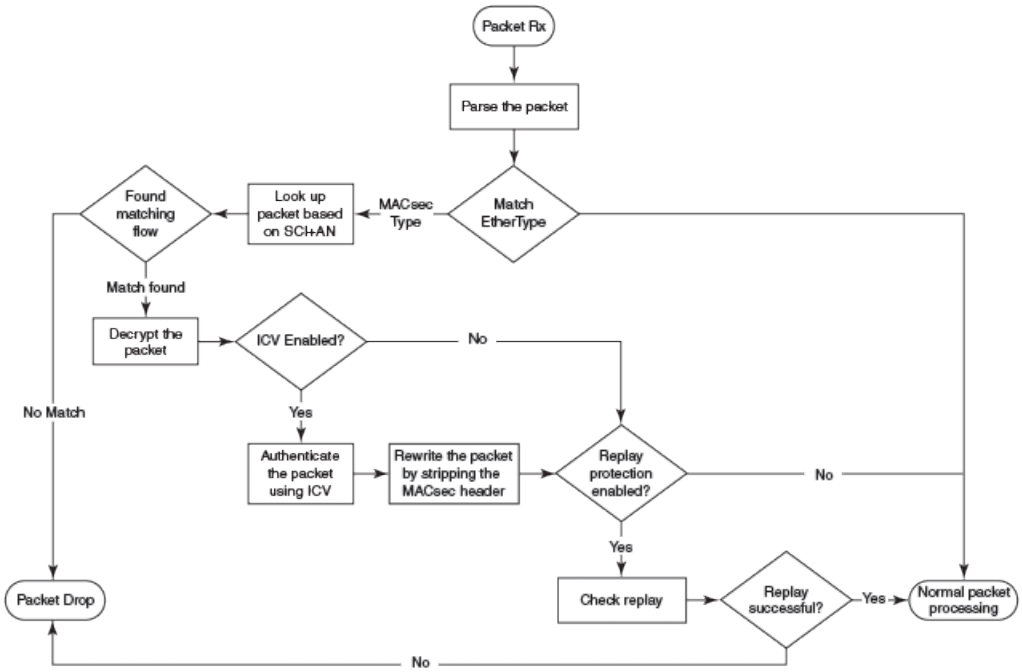


Processing Incoming Frames

FastIron hardware processes each MACsec frame received or transmitted based on the information in the MACsec security tag.

The FastIron device first confirms the Ethertype on incoming frames as MACsec and then processes incoming MACsec frames as illustrated in the following figure.

FIGURE 4 MACsec Incoming Frames

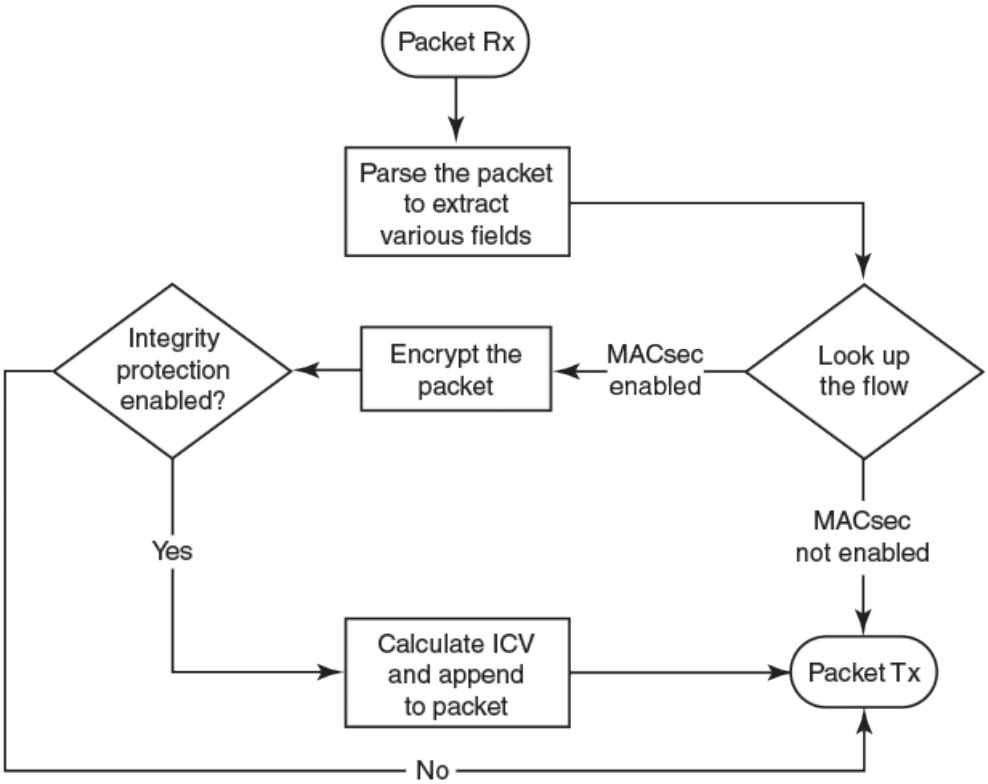


Processing Outgoing Frames

The FastIron device parses each outgoing frame and, if MACsec is enabled, processes the outgoing MACsec frame to apply configured MACsec options.

The following figure shows how the device applies configured MACsec options before transmitting the frames.

FIGURE 5 MACsec Outgoing Frames



Configuring MACsec

Although the MACsec configuration options outlined in this section are always visible, they cannot be applied unless an active license is present on the device and MACsec is enabled. MACsec licenses are required on a per-device basis. Each device in a stack requires a separate MACsec license.

These steps are required to configure MACsec security on a link or a group of connected ports.

1. Enter the dot1x-mka level from the global configuration level, and enable MACsec for the device.
2. Configure the MACsec Key Agreement (MKA) group.
3. Configure required parameters for the group, including frame validation, confidentiality, replay protection or delay protection, and actions to be taken when MACsec requirements are not met.
4. Create and configure an MKA keychain.
5. Enable MKA on each participating interface.
6. Apply the configured MKA group on the participating interface.

NOTE
If an MKA group is not applied to an enabled MACsec interface, or if parameters within the applied group have not been configured, default values are applied to the interface. Configured parameters are visible in **show** command output; default parameters are not always visible. Refer to the *RUCKUS FastIron Command Reference Guide* for default values for each command.

7. Apply the previously configured MKA keychain to the MACsec interface.

   NOTE
   As an alternative, you can configure the Connectivity Association Key (CAK) and Connectivity Association Key Name (CKN) on each interface. However, pre-shared key configuration and MKA keychain configuration are not allowed on the same interface.

Enabling MACsec and Configuring Group Parameters

Enable MACsec globally on the device, and configure the MACsec Key Agreement (MKA) group before configuring MACsec security features for the group.

A valid license must be installed on the device before MACsec can be configured.

1. At the global configuration level, enter the **dot1x-mka-enable** command to enable MACsec on the device.

   ```
   device# configure terminal
   device(config)# dot1x-mka-enable
   device(config-dot1x-mka)#
   ```

   MACsec is enabled, and the device is placed at the dot1x-mka configuration level.

   NOTE
   When MKA is disabled, all the ports are brought to a down state. You must manually enable the ports again to bring the ports back up.

2. Enter the **mka-cfg-group** command followed by a group name to create a group.

   ```
   device# configure terminal
   device(config)# dot1x-mka
   device(config-dot1x-mka)# mka-cfg-group test1
   device(config-dot1x-mka-group-test1)#
   ```

   The group is created, and the device is placed at the group configuration level.

Next: At the group configuration level, set the key-server priority, and define MACsec security features to be applied to interfaces.

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 148 for an overview of enabling and configuring MACsec features.

At the dot1x-mka group configuration level, enter the **key-server-priority** command, and specify a value from 0 through 255 to define the key-server priority.

```
device# configure terminal
device(config)# dot1x-mka-enable
```

```
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)# key-server-priority 20
```

In this example, the key-server priority is set to 20 for the MKA group test1.

Next: Configure MACsec integrity and encryption for the group.

Configuring MACsec Integrity and Encryption

To ensure point-to-point integrity, MACsec computes an Integrity Check Value (ICV) on the entire Ethernet frame using the designated cipher suite. The designated cipher suite is also used for encryption.

MACsec adds the ICV to the frame before transmission. The receiving device recalculates the ICV and checks it against the computed value that has been added to the frame. Because the ICV is computed on the entire Ethernet frame, any modifications to the frame can be easily recognized.

By default, both encryption and integrity protection are enabled.

MACsec encrypts traffic between devices at the MAC layer and decrypts frames within participating networked devices. MACsec uses the Galois/Counter Mode Advanced Encryption Standard 128 or 256 (GCM-AES-128 or GCM-AES 256) cipher suite to encrypt data and to compute the ICV for each transmitted and received MACsec frame.

MACsec also encrypts the VLAN tag and the original Ethertype field in the Layer 2 header of the secured data. When initial bytes in a secure data packet must be transparent, a confidentiality offset of 30 or 50 bytes can be applied.

NOTE
Refer to Configuring MACsec on page 148 for an overview of enabling and configuring MACsec features.

1. At the dot1x-mka group configuration level, enter the **macsec cipher-suite** command with one of the available options:
   - **gcm-aes-128**: Enables encryption and integrity checking using the GCM-AES-128 cipher suite.
   - **gcm-aes-128 integrity-only**: Enables integrity checking without encryption.
   - **gcm-aes-256**: Enables encryption and integrity checking using the GCM-AES-256 cipher suite.
   - **gcm-aes-256 integrity-only**: Enables integrity checking without encryption.

   In the following example, MACsec 128-bit encryption has been configured as a group test1 setting. By default, the ICV integrity check is also enabled, no matter which cipher suite you use.

   ```
   device# configure terminal
   device(config)# dot1x-mka
   device(config-dot1x-mka)# mka-cfg-group test1
   device(config-dot1x-mka-group-test1)# macsec cipher-suite gcm-aes-128
   ```

   In the following example, MACsec has been configured for integrity protection only, without encryption.

   ```
   device# configure terminal
   device(config)# dot1x-mka
   device(config-dot1x-mka)# mka-cfg-group test1
   device(config-dot1x-mka-group-test1)# macsec cipher-suite gcm-aes-128 integrity-only
   ```

   NOTE
   The **no 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 148 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 out-of-order 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 148 for an overview of enabling and configuring MACsec features.

At the dot1x-mka group configuration level, enter the **macsec replay-protection** command with one of the available modes:

- **strict**: Frames must be received in exact incremental sequence.
- **out-of-order** *window size*: Frames are accepted out of order within the designated window size. The maximum window size is 2147483648.
- **disable**: Frames are not validated.

In the following example, MACsec 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, MACsec replay protection is enabled for group test1. Frames are accepted out of order within the designated window size (100).

```
device# configure terminal
device(config)# dot1x-mka
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)# macsec replay-protection out-of-order window-size 100
```

Once you have configured the desired MKA group settings, these settings can be applied to specific interfaces.

Configuring Data-Delay Protection

MACsec data-delay protection allows MKA participants to ensure that the data frames protected by MACsec are not delayed by more than two seconds.

Each MACsec peer uses the MACsec Key Agreement (MKA) Protocol Data Unit (MKPDU) to communicate the lowest acceptable packet number. When a peer receives MACsec data with a packet number value less than the lowest acceptable packet number, MACsec increments the Delay Packet counters.

By default, the data-delay protection feature is disabled. Configuring the **macsec delay-protection** command under MKA group settings and attaching the group to a MACsec interface enables the data-delay protection feature on that interface.

NOTE
MACsec replay protection must be disabled when MACsec data-delay protection is configured.

NOTE
Refer to Configuring MACsec on page 148 for an overview of enabling and configuring MACsec features.

At the dot1x-mka group configuration level, enter the **macsec delay-protection** command.

In the following example, data-delay protection is enabled for group test1. Frames are protected when they are received with a delay of two seconds or less.

```
device# configure terminal
device(config)# dot1x-mka
device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka)# macsec replay-protection disable
device(config-dot1x-mka-group-test1)# macsec delay-protection
```

Once you have configured the desired MKA group settings, these settings can be applied to specific interfaces.

Enabling and Configuring Group Interfaces for MACsec

After MACsec is enabled for the device, each MACsec interface must be individually enabled, and a configured set of parameters must be applied.

1. To enable MACsec, at the dot1x-mka configuration level, enter the **enable-mka** command, and specify the interface as *unit/slot/port*.

   In the following example, Ethernet port 1 on slot 2 of unit 2 is enabled for MACsec security.

   ```
   device# configure terminal
   device(config)# dot1x-mka
   device(config-dot1x-mka)# enable-mka ethernet 2/2/1
   device(config-dot1x-mka-2/2/1)#
   ```

   NOTE
   The following output is displayed if there is no MACsec license present on the device.

   ```
   device(config-dot1x-mka)# enable-mka ethernet 2/2/1
   Error: No MACsec License available for the port 2/2/1. Cannot enable MACsec !!!
   Error: MKA cannot be enabled on port 2/2/1
   ```

2. At the dot1x-mka interface configuration level, enter the **mka-cfg-group** command, and specify the MKA group configuration to apply to the interface.

   ```
   device(config-dot1x-mka-2/2/1)# mka-cfg-group test1
   ```

3. This step is an option to configuring the pre-shared key as described later. At the dot1x-mka interface configuration level, enter the **mka-keychain** command, and specify the previously configured keychain to apply to the interface.

   ```
   device(config-dot1x-mka-2/2/1)# mka-keychain macsec1
   ```

In the following example, MACsec options configured for "group test1" and MKA keychain "macsec1" are applied to the enabled interface.

```
device# configure terminal
device(config)# dot1x-mka
device (config-dot1x-mka)# enable-mka ethernet 2/2/1
device(config-dot1x-mka-2/2/1)# mka-cfg-group test1
device(config-dot1x-mka-2/2/1)# mka-keychain macsec1
device(config-dot1x-mka-2/2/1)# end
device#
```

Configuring the Pre-shared Key

NOTE
Refer to Configuring MACsec on page 148 for an overview of enabling and configuring MACsec features.

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. The key can be configured directly on each MACsec interface, or you can configure a set of keys as an MKA keychain and apply the keychain to each interface. MKA keychain configuration is described in Creating and Configuring an MKA Keychain on page 281.

NOTE
Pre-shared key configuration and 'mka-keychain' configuration are not allowed on the same interface. If you have already configured and applied an MKA keychain to the MACsec interface, this task is not required.

At the dot1x-mka-interface configuration level, enter the **pre-shared-key** command followed by the *key-id*, the keyword **key-name**, and a hex string to define and name the pre-shared key.

The requirements for the parameters are as follows:

- *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 8-bit multiples).

In the following example, the pre-shared key with the hex value beginning with "135bd758b" and the key name beginning with "96437a93" are applied to interface 1/3/2.

```
device# configure terminal
device(config)# dot1x-mka
device (config-dot1x-mka)# enable-mka ethernet 1/3/2
device(config-dot1x-mka-1/3/2)# pre-shared-key 135bd758b0ee5c11c55ff6ab19fdb199 key-name
96437a93ccf10d9dfe347846cce52c7d
```

Enable and configure each MACsec interface. Configure the same pre-shared key (CAK) on the interfaces between which a secure channel can be established.

Sample MACsec Configuration

The following example shows how to enable MACsec, configure general parameters, enable and configure interfaces, and create and assign a keychain module to an interface. The keychain module must also be assigned to peer interfaces.

```
device# configure terminal
device(config)# dot1x-mka-enable

device(config-dot1x-mka)# mka-cfg-group test1
device(config-dot1x-mka-group-test1)# key-server-priority 5
device(config-dot1x-mka-group-test1)# macsec cipher-suite gcm-aes-128
device(config-dot1x-mka-group-test1)# macsec confidentiality-offset 30
device(config-dot1x-mka-group-test1)# macsec frame-validation strict
device(config-dot1x-mka-group-test1)# macsec replay-protection strict
device(config-dot1x-mka-group-test1)# exit
device(config-dot1x-mka)# exit

device(config)# keychain macsec1 mka
device(config-keychain-mka-macsec1)# key-id 1
device(config-keychain-mka-macsec1-key-1)# password ........
device(config-keychain-mka-macsec1-key-1)# authentication-algorithm aes-256-cmac
device(config-keychain-mka-macsec1-key-1)# send-lifetime start 02-14-2022 01:01:01 end 03-14-2022 06:59:00
device(config-keychain-mka-macsec1-key-1)# end

device# configure terminal
device(config)# dot1x-mka
device(config-dot1x-mka)# enable-mka ethernet 1/3/2

device(config-dot1x-mka-1/3/2)# mka-cfg-group test1
device(config-dot1x-mka-1/3/2)# mka-keychain macsec1
device(config-dot1x-mka-1/3/2)# end
device#
```

Displaying MACsec Information

Use MACsec **show** commands to display information on MACsec for a device, group, or individual interface. MACsec **show** commands can be used to display configuration information, to report on MACsec sessions that are currently active on a device, or to monitor MACsec statistics on a particular interface.

Displaying MACsec Configuration Details

You can display configuration information for all MACsec groups on a device, or you can display details for a particular group.

1. In privileged EXEC, global configuration, or dot1x-mka interface mode, use the **show dot1x-mka config** command to display MACsec configuration details for the device.

   In the following example, MACsec parameters are displayed for the device and all groups configured on it. Specific MACsec interfaces are displayed as well as the pre-shared key for each interface.

   NOTE
   The pre-shared key is displayed as an encrypted value, not in plain text.

```
device(config)# show dot1x-mka config
dot1x-mka-enable
mka-cfg-group group1
  key-server-priority 20
  macsec frame-validation check
  macsec confidentiality-offset 30
  macsec cipher-suite gcm-aes-128
  macsec-replay protection out-of-order window-size 100
  enable-mka ethernet 1/3/2
mka-cfg-group group1
  pre-shared-key xxxxxxxxxxxxxxxxxxxxxxxx key-name 96437a93ccf10d9dfe3478460cce5132
enable-mka ethernet 1/3/6
  mka-cfg-group group1
  pre-shared-key xxxxxxxxxxxxxxxxxxxxxxxx key-name 96437a93ccf10d9dfe3478460cce51321
```

2. In privileged EXEC, global configuration, or dot1x-mka interface mode, enter the **show dot1x-mka config-group** command to display information for all configured groups. Add a group name to the command to narrow the information displayed to one group.

   The following example displays information for MKA group test1.

```
device(config)# show dot1x-mka config-group test1
 mka-cfg-group test1
  key-server-priority 5
  macsec cipher-suite gcm-aes-128 integrity-only
  macsec confidentiality-offset 30
  macsec frame-validation strict
  macsec replay-protection 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.

3. In privileged EXEC, global configuration, or dot1x-mka interface mode, use the **show keychain mka** command to display details for configured MACsec keychains. For additional details on a specific keychain, use the **show keychain name** command.

   The following example shows that one MKA keychain named "sample" has been configured.

```
device# show keychain mka

Keychain  : sample
Tolerance : 0
------------------------------------------------------------------------------
  Key-id  |    Algo    | SendActive | SendTimer | AcceptActive | AcceptTimer
------------------------------------------------------------------------------
 100       aes-128-cmac Yes(GMT+00)      -        No(GMT+00)          -
```

   The following example uses the **show keychain name** command followed by the keychain name displayed in the previous example to display additional details for the specified keychain.

```
device# show keychain name sample
Keychain: sample
  Tolerance: 0
  Key-id   : 100
      AuthAlgorithm: aes-128-cmac
      Key-String   : *******
      Send Lifetime:-
          Start    : 02-16-2022 04:05:00 End         : Infinite
          Active   : Yes                 TimeToExpire: Infinite
          Timezone : GMT+00
```

Displaying Information on Current MACsec Sessions

You can display MACsec session activity for an interface, including the pre-shared key name, the most recent SAI information, and a list of peers.

1. For a quick overview of current MACsec sessions, enter the **show dot1x-mka sessions brief** command in privileged EXEC, global configuration, or dot1x-mka interface mode.

```
device(config)# show dot1x-mka sessions brief

Port    Link-Status  MKA-Status  Key-Server  Negotiated Capability

1/3/2   Down         Pending   ---      ---
1/3/3   Up           Secured   No       Integrity, Confidentiality with Off. 30
1/3/4   Up           Secured   No       Integrity, Confidentiality with Off. 30
```

2. To display full details on current MACsec sessions, in privileged EXEC, global configuration, or dot1x-mka interface mode, enter the **show dot1x-mka sessions ethernet** command followed by the interface identifier.

```
device(config)# show dot1x-mka sessions ethernet 1/3/3

Interface                : 1/3/3

  MACsec Status          : Secured
  DOT1X-MKA Enabled      : Yes
  DOT1X-MKA Active       : Yes
  Key Server             : No

Configuration Status:
  Enabled                : Yes
  Capability             :  Integrity,  Confidentiality
  Desired                : Yes
  Protection             : Yes
  Frame Validation       : Disable
  Replay Protection      : Strict
  Replay Protection Size : 0
  Cipher Suite           : GCM-AES-128
  Key Server Priority    : 20

  Local SCI              : 748ef8344a510082
  Member Identifier      : 802ed0536fcafc43407ba222
  Message Number         : 8612

Secure Channel Information:
  Latest SAK Status      : Rx & Tx
  Latest SAK AN          : 0
  Latest KI              : d08483062aa9457e7c2470e300000001
  Negotiated Capability  : Integrity, Confidentiality with offset 30

Peer Information:
State        Member Identifier           Message Number    SCI                Priority
-----        -----------------           --------------    ----------------   --------
Live         d08483062aa9457e7c2470e3              8527    748ef83443910082         20
```

Displaying MKA Protocol Statistics for an Interface

You can display a report on MKA protocol activity for a particular interface.

In privileged EXEC, global configuration, or dot1x-mka interface mode, enter the **show dot1x-mka statistics ethernet** command to display MKA protocol statistics for the designated interface.

```
device(config-dot1x-mka-1/3/3)# show dot1x-mka statistics ethernet 1/3/3

Interface                : 1/3/3

MKA in Pkts              : 8585
MKA in SAK Pkts          : 1
MKA in Bad Pkts          : 0
MKA in Bad ICV Pkts      : 0
MKA in Mismatch Pkts     : 0
MKA out Pkts             : 8687
MKA out SAK Pkts         : 0
Number of SAK            : 1
```

Displaying MACsec Secure Channel Activity for an Interface

You can display currently enforced MACsec capabilities for a specific interface, along with secure channel statistics.

1. In privileged EXEC mode, enter the **clear macsec statistics** command for the designated interface.

   Results of the previous **show macsec statistics** command are removed.

2. In privileged EXEC, global configuration, or dot1x-mka interface mode, enter the **show macsec statistics** command to display information on MACsec configuration and secure channel activity for a particular interface.

```
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 161
- Port MAC Security Configuration 162
- Configuring Port MAC Security 163
- Displaying Port MAC Security Information 164
- Clearing Restricted MAC Addresses and Port Security Violation Statistics 165

Port MAC Security Overview

Port MAC security (PMS) feature allows you to configure the device to learn a limited number of secure MAC addresses on an interface. The interface forwards only those packets with source MAC addresses that match these secure addresses.

The secure MAC addresses can be specified statically or learned dynamically. If the device reaches the maximum limit for the number of secure MAC addresses allowed on the interface and if the interface receives a packet with a source MAC address that is different from any of the secure learned addresses, it is considered a security violation. MAC addresses that are learnt beyond the configured PMS maximum threshold value are considered as restricted MAC addresses. When a security violation occurs, an action is taken according to one of three configurable modes, as summarized in the following table.

When a security violation occurs, a Syslog entry and an SNMP trap are generated.

TABLE 16 PMS Violation Actions/Modes

Violation Actions/Modes	Description
Protect	This is the default PMS violation action. Allows packets from secure addresses and drops all other packets. In the protect mode, the port never gets shut down. SNMP trap and Syslog are generated when the port enters and leaves the protect mode.
Restrict	Drops packets from the restricted MAC addresses and allows packets from the secure addresses. The maximum number of MAC addresses that can be restricted is 128. If the number of restricted MAC addresses exceeds 128, the port is shut down. In this mode, manual intervention is required to bring up the port that is forced to shut down after the security violation. SNMP trap and Syslog are generated for each restricted MAC address. Also, when a port is shutdown a separate trap and Syslog are generated to indicate the same.
Shutdown	Shuts down the port upon detection of first restricted MAC address. The shutdown time which serves as a recovery interval, brings up the port within a configured time without any manual intervention. SNMP trap and Syslog are generated when the port shuts down.

The secure MAC addresses are flushed when an interface is disabled and re-enabled on ICX devices. The secure addresses can be kept secure permanently (the default), or can be configured to age out, at which time they are no longer secure. You can configure the device to automatically save the secure MAC address list to the startup-config file at specified intervals, allowing addresses to be kept secure across system restarts.

Local and Global Resources Used for Port MAC Security

The port MAC security feature uses a concept of local and global resources to determine how many MAC addresses can be secured on each PMS-enabled port. In this context, a resource is the ability to store one secure MAC address entry. When an interface has secured enough MAC addresses to reach its configured limit for local resources, it uses the global resources to secure additional MAC addresses.

When the port MAC security feature is enabled on an interface, the interface can store one secure MAC address (default). The maximum number of secure MAC addresses that can be secured using local resources is 64 MAC addresses per PMS-port. Besides the maximum of 64 local resources

available for an interface, additional 8192 global resources are shared among all PMS-enabled interfaces on the device by default. The default system value of the global resources can be changed using the **system-max pms-global-pool** command. Global resources are shared among all the interfaces on a first-come, first-served basis. The maximum number of MAC addresses any single interface can secure is 64 (the maximum number of local resources available to the interface), plus the number of global resources not allocated to other interfaces.

Configuration Considerations for Port MAC Security

The following limitations apply to the port MAC security (PMS) feature:

- Applies only to Ethernet interfaces.
- PMS is not supported on PVLAN ports.
- Unknown unicast traffic is flooded out of port with maximum secure MAC learnt on removing the ACL.
- Not supported on static trunk group members or ports that are configured for link aggregation.
- Not supported on 802.1X authentication-enabled ports.
- The SNMP trap generated for restricted MAC addresses indicates the VLAN ID associated with the MAC address, as well as the port number and MAC address.
- Not supported on ports that have MAC authentication enabled.
- The first packet from each new secure MAC address is dropped if secure MAC addresses are learned dynamically.
- Violated MAC movement is not supported.
- Three MAC addresses must be allocated in port security when connecting IP Phone with PC connected to it. Because, the VOIP phone initially connects untagged. After it gets its configuration and the voice VLAN from FastIron device, it connects tagged with the voice VLAN. The tagged and untagged for the same MAC address are considered as two different entries. Then the PC connected to the phone gets its MAC address registered untagged, consuming a total of three entries.

Secure MAC Movement

If you move a connected device that has MAC address configured as secure on one port to another port, the FastIron device connects through the new port without waiting for the MAC address to age out on the previous port. This MAC movement feature is supported when the connected device moves from a secure port to another secure or non-secure port.

MAC movement feature is not supported in the following cases:

- MAC address is permanently secured to a port with **age 0** command.
- MAC address causes a MAC security violation on the previous port.

NOTE
Excessive Syslog messages are expected when MAC movement happens on permanently secured MAC address. Use the **no logging buffered warnings** command to suppress warning Syslogs.

Port MAC Security Configuration

To configure the port MAC security feature, perform the following tasks:

- Enable the port MAC security feature
- Set the maximum number of secure MAC addresses for an interface
- Set the port security age timer
- Specify secure MAC addresses

- Configure the device to automatically save secure MAC addresses to the startup-config file
- Specify the action taken when a security violation occurs

NOTE
When using the **absolute** option to age out MAC addresses on timer expiry, make sure that the age timer value is sufficient. Avoid using a very short timer expiry with the **absolute** option, as the value may be in conflict with other timer settings and may cause performance problems in the network. For example, a one-minute timer expiry will cause MAC addresses to be flushed every minute. As a result, operational (enable/disable) loops and packet flooding may occur following a security violation, which by default causes a port to be disabled for one minute.

Configuring Port MAC Security

Port MAC security can be configured globally or on a specific interface.

By default, the MAC port security feature is disabled on all interfaces.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter port security configuration mode.

```
device(config)# port security
```

3. Enable port MAC security globally on the device.

```
device(config-port-security)# enable
```

4. Set the maximum number of secure MAC addresses.

```
device(config-port-security)# maximum 10
```

   By default, when MAC port security is enabled, an interface can store one secure MAC address. A maximum of 64 local resources can be allocated to each interface.

5. (Optional) From the global configuration mode, configure the maximum number of global resources shared among all interfaces on the device to store secure MAC addresses.

```
device(config)# system-max pms-global-pool 800
```

   By default, 8192 global resources are shared among all interfaces on the device which is also the maximum global resources supported. The global resources are in addition to the local resources allocated to each interface . The maximum number of MAC addresses any single interface can secure is 64 (the maximum number of local resources allocated to the interface), plus the number of global resources available.

6. Specify a secure MAC address.

```
device(config-port-security)# secure-mac-address 000.0018.747c
```

7. Set the optional port security age timer. By default, learned MAC addresses stay secure indefinitely.

```
device(config-port-security)# age 60 absolute
```

   In this example, secure MAC addresses are immediately timed out after 60 minutes. If the absolute option is not configured, the secure MAC addresses time out when the hardware MAC age timer expires.

8. Set the optional time interval when learned secure MAC addresses are saved to the startup configuration.

```
device(config-port-security)# autosave 20
```

   In this example, learned secure MAC addresses are saved to the startup configuration every 20 minutes.

9. Configure the action that must be taken when a port security violation occurs. Select one of the following configurable modes that specifies the violation action.
   - Configure the protect mode to drop all packets which are not from secure MAC addresses.

```
device(config-port-security)# violation protect
```

     This is the default PMS violation action. In the protect mode, the port never gets shut down.
   - Configure the restrict mode to drop packets from violated MAC address and allow packets from secure addresses.

```
device(config-port-security)# violation restrict
```

     When the **restrict** option is used, maximum number of MAC addresses that can be restricted is 128. If the number of violated MAC addresses exceeds 128, the port will be shut down. In this mode, manual intervention is required to bring up the port that is forced to shut down after the security violation. Aging for restricted MAC addresses is done in software. There can be a worst case inaccuracy of one minute from the specified time. The restricted MAC addresses are denied in hardware.
   - Configure the shutdown mode to disable the port upon detection of first violated MAC address. for a specified amount of time, in minutes, when a security violation occurs.

```
device(config-port-security)# violation shutdown
```

     The shutdown time which serves as a recovery interval, brings up the port within a configured time without any manual intervention. The default is value is 0 which shuts down the port permanently when a security violation occurs.

In the following example, port MAC security is configured globally.

```
device# configure terminal
device(config)# port security
device(config-port-security)# enable
device(config-port-security)# maximum 10
device(config-port-security)# secure-mac-address 000.0018.747c
device(config-port-security)# age 60 absolute
device(config-port-security)# autosave 20
device(config-port-security)# violation restrict 100
```

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

The following example displays the port security settings for an individual port.

```
device# show port security ethernet 1/7/11

Port   Security Violation Shutdown-Time Age-Time Max-MAC
------ -------- --------- ------------- -------- -------
1/7/11 disabled  shutdown            10       10       1
```

The following example lists the secure MAC addresses configured on the device.

```
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
```

The following example displays port security statistics for an individual port.

```
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
```

Clearing Restricted MAC Addresses and Port Security Violation Statistics

You can clear restricted MAC addresses and violation statistics on all ports or from individual ports. The following steps are optional and can be used in any order.

1. Clear all restricted MAC addresses globally.

   ```
   device# clear port security restricted-macs all
   ```

   To clear restricted MAC addresses on a specific port, enter the following command.

   ```
   device# clear port security restricted-macs ethernet 5/1/5
   ```

2. Clear port security violation statistics globally.

   ```
   device# clear port security statistics all
   ```

   To clear port security violation statistics on a specific port, enter the following command.

   ```
   device# clear port security statistics ethernet 1/1/5
   ```

Flexible Authentication

- Flexible Authentication Overview 167
- 802.1X Authentication 177
- MAC Authentication 181
- RADIUS Attributes for Authentication and Accounting 186
- Configuring ICX Vendor-Specific Attributes on the RADIUS Server 189
- Support for the RADIUS User-Name Attribute in Access-Accept Messages 191
- Dynamic VLAN Assignment 191
- Dynamic ACLs in Authentication 196
- Support for IP Source Guard Protection 199
- Configuring Flexible Authentication 200
- Displaying Authentication Information 208
- Clearing Authentication Details 212

Flexible Authentication Overview

Flexible authentication combines MAC authentication and 802.1X authentication as a single authentication procedure.

NOTE
Flexible authentication is not supported in private VLANs.

NOTE
The term "client" is used to indicate the user or device that is going through authentication.

In a network, many types of clients may gain access and use the network resources. Such networks cannot be left unrestricted due to security concerns. There must be a mechanism to enforce authentication of the clients before allowing access to the network. In addition, a single authentication method may not be compatible for all the clients that support different authentication methods. In such cases, it is not feasible to assign separate ports with specific authentication methods for different types of clients. 802.1X authentication and MAC authentication, and a combination of both, provide strong, yet flexible methods to validate the clients and prevent unauthorized clients from gaining access to the network. If the authentication succeeds, the client (the MAC address of the client) is moved to a VLAN returned by the RADIUS server, and the policies returned by the RADIUS server are applied.

RUCKUS ICX devices support the IEEE 802.1X standard for authenticating clients attached to data ports. Using 802.1X, you can configure an ICX device to grant or deny access to a port based on information supplied by a client to an authentication server.

When a user logs in to a network that uses 802.1X, the RUCKUS ICX device grants or denies access to network services after the user is authenticated by an authentication server. The user-based authentication in 802.1X provides an alternative to granting network access based on a user IP address, MAC address, or subnetwork.

MAC authentication is another mechanism by which incoming traffic originating from a specific MAC address is switched or forwarded by the device only if an authentication server successfully authenticates the source MAC address. The MAC address itself is used as the username and password for authentication; the user does not need to provide a specific username and password to gain access to the network. If authentication for the MAC address is successful, traffic from the MAC address is forwarded in hardware.

Flexible authentication provides a means to set the sequential order in which 802.1X authentication and MAC authentication methods need to be executed. If both authentication methods are enabled on the same port, by default, the authentication sequence is set to perform 802.1X authentication followed by MAC authentication. Both the 802.1X authentication and MAC authentication methods must be enabled at the global and interface levels on the same port to execute Flexible authentication. Flexible authentication facilitates multiple authentication methods to

validate a client using a single configuration on the same port. Thus, different clients that support different types of authentication can be authenticated using a single configuration.

After successful authentication, different policies can be applied to restrict the way the client accesses network resources. VLAN policies, phone policies, and ACL policies can be enforced using VLAN assignment, ACL assignment, and phone-specific information to provide different levels of service to the client and to control the destination of the client.

The following table shows ICX features that have been tested for compatibility with various Network Access Control (NAC) applications.

TABLE 17 Features Tested for Compatibility with Various NAC Applications

Feature	Free RADIUS	Aruba ClearPass	RUCKUS Cloudpath	Cisco ISE
802.1X authentication	Yes	Yes	Yes	Yes
MAC authentication	Yes	Yes	Yes	Yes
Dynamic VLAN assignment	Yes	Yes	Yes	Yes
Dynamic ACL assignment	Yes	Yes	Yes	Yes
External web authentication	Not applicable	Yes	Yes	Yes (Release 2.1)
Change of Authorization (CoA)	Yes	Yes	Yes	Yes (Release 2.1)

NOTE
Refer to the *RUCKUS FastIron Features and Standards Support Matrix* for the list of supported platforms.

MAC VLANs

Traditional VLANs associate ports as untagged or tagged. A port can only belong to a single VLAN as untagged, yet it can be part of multiple VLANs as a tagged member.

Packets received at the port are classified into VLANs based on the VLAN tag carried in the packet (tagged VLANs); otherwise, the packet is classified with port untagged VLAN.

Using a MAC VLAN is a way of classifying the packets based on the source MAC address with the help of hardware maintaining the MAC-VLAN table.

After successful authentication, VLANs are dynamically assigned based on the client profiles configured on the RADIUS server. The ICX device then associates the port with the dynamic VLAN only for the specific client MAC address. With this option, a port can belong to multiple VLANs as a MAC-VLAN member. All such packets coming from the respective clients are untagged and are classified into appropriate VLANs when they are received on the ICX device. This makes the port look as if it is part of multiple untagged VLANs.

In summary, after successful authentication, the RADIUS server returns the details of the VLAN where the client should belong. The client (the MAC address of the client) is moved to this VLAN as a MAC-VLAN member. The client is removed from the corresponding VLAN when the client logs out, the port goes down, or the MAC address ages out.

Data VLAN Requirements for Flexible Authentication

For deploying Flexible authentication, VLANs such as the auth-default VLAN, restricted VLAN, critical VLAN, and guest VLAN are used for various success, failure, and timeout scenarios. The use of these VLANs provides network administrators more granular access control for different client scenarios.

Before authentication is enabled on a port, the port can belong to any VLAN, including the system default VLAN. The only restriction is that the port cannot be a part of any VLAN as untagged. After authentication is enabled on that port, the port becomes a part of the auth-default VLAN. When a VLAN is assigned after successful authentication, it is assigned to the client (to the MAC address of the client), not to the entire port. In reality, however, the port is added to the VLAN as a MAC-VLAN member.

When authentication succeeds, the client is moved to the VLAN returned by the RADIUS server. When the RADIUS authentication fails or the RADIUS server is unavailable, the client is moved to the restricted VLAN or critical VLAN.

NOTE
A system default VLAN, reserved VLAN, or VLAN group cannot be used as the auth-default VLAN, RADIUS-assigned VLAN, restricted VLAN, critical VLAN, or guest VLAN. A system default VLAN cannot be used for Web authentication.

NOTE
A RADIUS-assigned VLAN can also serve as one or more of the following VLANs: restricted VLAN, critical VLAN, or 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 Flexible authentication. When any port is enabled for 802.1X authentication or MAC authentication, the client is moved to this VLAN by default. The auth-default VLAN is also used in the following scenarios:
  - When the RADIUS server does not return any VLAN information upon authentication, the client is authenticated and remains in the auth-default VLAN.
  - If RADIUS timeout occurs during the first authentication and the timeout action is configured as "Success", the client is authenticated in the auth-default VLAN. If the RADIUS server is not available during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN.
- Restricted VLAN: When an authentication fails, the port can be moved into a restricted VLAN instead of blocking the client completely. The port is moved to the configured restricted VLAN only if the authentication failure action is configured to place the port in a restricted VLAN using the **auth-fail-action** command in the authentication configuration mode. Otherwise, when the authentication fails, the client's MAC address is blocked in the hardware, which is the default action. A restricted VLAN can be configured using the **restricted-vlan** command in the authentication configuration mode.
- Critical VLAN: There may be times when the RADIUS server times out or is not available, resulting in timeout. This can happen the first time the client is authenticating or when the client reauthenticates. In such scenarios, if the authentication timeout action is specified as a critical VLAN using the **authentication timeout-action** command in the authentication configuration mode, the client is moved to the specified critical VLAN. A critical VLAN can be configured using the **critical-vlan** command in the authentication configuration mode.
- Guest VLAN: The guest VLAN is used when a client does not respond to dot1x requests for authentication. It is possible that the client does not support or have the dot1x supplicant loaded. In such a scenario, the client is moved to the guest VLAN to have access to the network with default privileges. From the guest VLAN, the client can download the supplicant.

  NOTE
  The same VLAN can be specified as the guest VLAN, restricted VLAN, and critical VLAN.

Voice VLAN Requirements for Flexible Authentication

Voice VLAN configuration facilitates the continued functioning of Voice over IP (VoIP) phones when external server authentication or authorization fails.

A voice VLAN is configured at the global level (using the **voice-vlan** command in authentication configuration mode) or at the local level (using the **authentication voice-vlan** command in interface configuration mode).

When a local voice VLAN is configured, it overrides the global voice VLAN configuration. The global voice VLAN is used only when a voice VLAN is not configured on the port.

A voice VLAN is used for voice communication by IP phones in the following ways:

- Authentication Success: When a RADIUS server authenticates an IP phone and the server does not specify a VLAN, the device is placed in the auth-default VLAN and also added to the voice VLAN (as a tagged member). The auth-default VLAN is used for initial authentication, learning the IP address, and so on, while the voice VLAN is used for voice traffic.
- Authentication Failure: When a phone fails authentication by a RADIUS server, the phone is blocked in hardware by default. To ensure that an IP phone continues to operate and that both data and voice traffic from the IP phone are appropriately forwarded, use the **auth-fail-action** command to set the fail action to **restricted-vlan**, and specify the **voice voice-vlan** option. This causes the device to be placed in the restricted VLAN for data traffic and the voice VLAN (as a tagged member) for voice traffic.
- Authentication Timeout: When a RADIUS server is not reachable, the phone cannot be authenticated, in which case several timeout-action options are available: treat as success, treat as failure, and treat as critical. Success and Failure cases work as explained previously. The critical-VLAN case works in a similar way to the restricted-VLAN case. To ensure that an IP phone continues to operate and that both data and voice traffic from the IP phone are appropriately forwarded, use the **auth-timeout-action** command to set the timeout action to **critical-vlan**, and specify the **voice voice-vlan** option. This causes the device to be placed in the critical VLAN for data traffic and the voice VLAN (as a tagged member) for voice traffic.

Authentication Modes

Flexible authentication supports the following four modes to address the requirements of different usage models. The authentication-enabled ports can be configured to be members of any mode using the **auth-mode** command at the global level or using the **authentication auth-mode** command from the interface.

- Multiple Untagged: Different clients on the same port can be placed in different untagged VLANs. Each client can be assigned a different VLAN by the RADIUS server. Some clients who fail can move to a restricted VLAN. Some clients who time out out can move to a guest VLAN and other VLANs.
- Single Untagged: This is the default auth-mode, where all the clients belong to one untagged VLAN only. This mode is the most common use case. When a hub and multiple clients are connected, the first client is moved to the RADIUS-assigned VLAN, and the subsequent clients are placed in the same VLAN. The subsequent authenticated clients are moved to the same VLAN even if RADIUS does not return any VLAN. If RADIUS returns different untagged VLANs, subsequent clients are blocked.
- Single Host: This mode allows authentication of only one host, and the status of the host is determined by the authentication. Access for all subsequent hosts is denied or blocked. In contrast, any connected IP phones are allowed access without authentication. The session exists only for the first host authenticated.
- Multiple Hosts: This mode allows authentication of the first device only, and the status of all other devices depends on the authentication status of the first device. This mode is typically used when the connecting device is a wireless AP that is authenticated by the ICX device, and the AP authenticates all pass-through wireless clients on the ICX device.

Dynamic assignment of VLANs in these modes varies depending on the format of the VLAN information in the RADIUS-returned Access-Accept message and whether the authentication is initiated by tagged or untagged ports. For more information, refer to Dynamic VLAN Assignment on page 191.

Tagged VM Client Support

Flexible authentication supports the authentication of VM clients that operate in tagged VLANs. When the port is tagged, the connected VMs are placed in the VM-tagged VLANs as expected. This arrangement requires the administrator to configure the ports to be tagged and to plan the network ahead of time, and the approach becomes static and inflexible for dynamic network usage changes.

The best alternative is to allow the tagged traffic from VMs to trigger authentication. When the port is not tagged, authenticate it, and dynamically tag the port in the required tagged VLANs as long as the VMs are in use. When the VM sessions on the ICX device expire for various reasons, the port is deleted from the respective tagged VLANs. This support is disabled by default and can be enabled at the interface level with the

authentication allow-tagged configuration command. This command is applicable only at the interface level and must be applied on all the interfaces where it is required.

Static Authentication with MAC Authentication Filters

Some devices may always need to be authenticated (included on a whitelist), and some may always have to be blocked (included on a blacklist). Device disposition can be determined at the time of authentication through the RADIUS server; however, for convenience, MAC authentication filters can be configured on ICX devices and applied to appropriate ports so that the ports are pre-authenticated (excluded from authentication based on MAC address).

The MAC authentication filter must be used when the RADIUS server itself is connected to an interface on which MAC authentication or 802.1X authentication is enabled. If a MAC authentication filter is not defined for the MAC addresses of the RADIUS server and applied on the interface, the RADIUS authentication process fails because the device drops all packets from the RADIUS server itself.

The MAC authentication filter is applied on an interface using the **authentication filter** command in interface configuration mode. A client can be authenticated in an untagged VLAN or tagged VLAN. If the MAC authentication filter has a tagged VLAN configuration, the clients are authenticated in the auth-default VLAN and the tagged VLAN provided in the authentication filter statement. The clients authorized in the auth-default VLAN allow both untagged and tagged traffic.

Authentication Actions

With Flexible authentication, success, failure, and timeout actions are applied for MAC authentication and 802.1X authentication.

Authentication Success Action

When the authentication order is set to perform 802.1X authentication followed by MAC authentication (the default Flexible authentication sequence), on 802.1X authentication, the client is authenticated, and the policies returned by the RADIUS server are applied.

When the authentication order is set to perform MAC authentication followed by 802.1X authentication, by default, 802.1X authentication is performed even if MAC authentication is successful. On successful 802.1X authentication, the client is authenticated, and the policies returned by the RADIUS server are applied.

Authentication Failure Action

A single failure action can be defined for both 802.1X authentication and MAC authentication. An administrator can take the following actions when there is an authentication failure:

- Block the client access (the default action): This blocks the client from accessing any network resource for a configured amount of time, after which the client can try authenticating again.
- Move the client to a restricted VLAN: This will happen only when the authentication server sends ACCESS-REJECT. It moves the client to a preconfigured restricted VLAN. Any access policies applied in that VLAN apply to this client.

Authentication Timeout Action

A single authentication timeout action can be specified for MAC authentication and 802.1X authentication timeouts with the RADIUS server.

A RADIUS timeout occurs when the ICX device does not receive a response from a RADIUS server within a specified time and after a certain number of retries. The time limit and number of retries can be manually configured using the **radius-server timeout** and **radius-server retransmit** commands. If the parameters are not manually configured, the ICX device applies the default value of 3 seconds with a maximum of 3 retries.

NOTE
The **radius-server retransmit** command is not applicable to RADIUS server TLS connections.

Administrators can control port behavior when a RADIUS timeout occurs by configuring a port on the ICX device to automatically pass or fail user authentication. A pass allows the client to continue with the VLAN and other policies. A fail blocks the client by default, unless a restricted VLAN or a default ACL is configured, in which case, the user is placed into a VLAN.

The following options are available:

- Failure (the default): This action blocks the client from accessing any network resource for a configured amount of time. If the failure action is configured as a restricted VLAN, the client is moved to the restricted VLAN.
- Success: The client is authenticated in the auth-default VLAN or in the previously authenticated VLAN, depending on the following conditions:
  - If RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the auth-default VLAN.
  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN.
  - If the RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the critical VLAN.
  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN.
- Critical VLAN: The client is moved to a preconfigured critical VLAN. Any access policies applicable to that VLAN apply to this client.

NOTE
Reauthentication is supported for auth-default, restricted, and critical VLANs. It is not supported for guest VLANs.

NOTE
The same VLAN can be specified as guest VLAN, restricted VLAN, and critical VLAN, if desired.

Reauthentication for the timed out clients that have been placed in the critical, restricted, or auth-default VLAN or the BLOCKED state (VLAN 4092) can be configured globally using the **authentication reauth-timeout** command. By default, the timeout is enabled and is set to 300 seconds.

Authentication Session Limits on an Interface

Flexible authentication allows multiple MAC addresses to be authenticated or denied on each interface.

By default, the number of MAC sessions that can be authenticated on a single interface is two. The maximum number of sessions can be changed using the **authentication max-sessions** command. The maximum number of authenticated MAC sessions on an interface depends on the ICX device and dynamic ACL assignments. If RADIUS assigns dynamic ACLs to at least one client on the interface, the maximum number of MAC sessions that can be authenticated is limited to 32 in all ICX devices.

If a dynamic ACL is not assigned to any of the clients on the interface, the maximum number of MAC addresses that can be authenticated varies depending on the ICX device as specified in the following table. System reload is not required for the changes to take effect. However, existing sessions on the interface are cleared for the changes to take effect.

TABLE 18 Maximum Number of Authenticated MAC Sessions per Port by ICX Device

Supported Platforms	Maximum Number of MAC Sessions per Port When None of the Clients Has Dynamic ACL	Maximum Number of MAC Sessions per Port When at Least One Client Has Dynamic ACL
ICX 8200	1024	32
ICX 7850	1024	32
ICX 7650	1024	32
ICX 7550	1024	32

The system limit for authenticated MAC sessions also varies and depends on the ICX device and dynamic ACL assignments.

TABLE 19 Maximum Number of Authenticated MAC Sessions per System (Standalone or Stack) by ICX Device

Supported Platforms	Maximum Number of MAC Sessions per System When None of the Clients Has Dynamic ACL	Maximum Number of MAC Sessions per System When at Least One Client Has Dynamic ACL
ICX 8200	1536	512
ICX 7850	1536	512
ICX 7650	1536	512
ICX 7550	1536	512

How Flexible Authentication Works

Flexible authentication can be configured at the global or interface level.

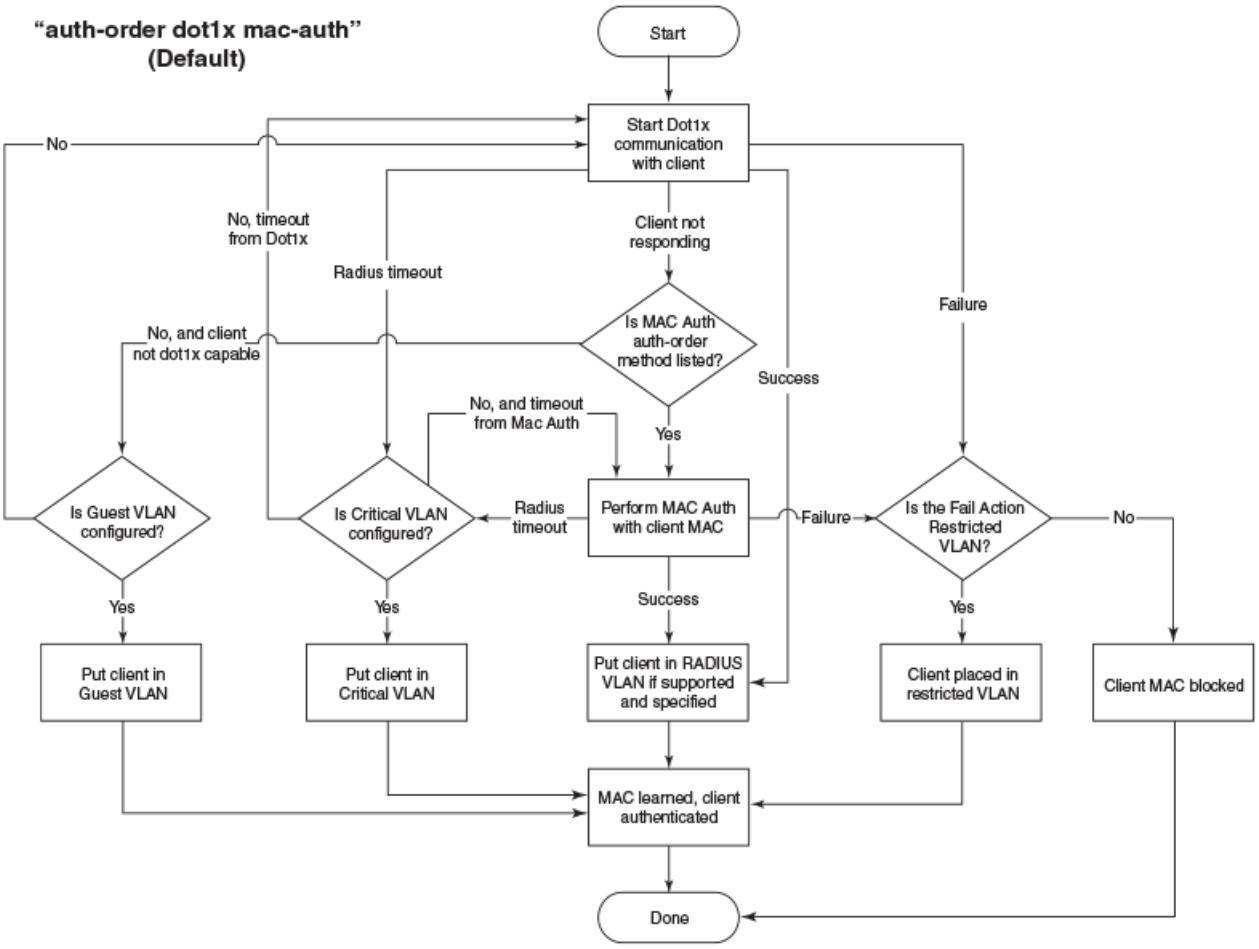
When only 802.1X authentication or MAC authentication is configured, the configured method is attempted. When authentication fails, the MAC address of the device is blocked (the default action) or is moved to a restricted VLAN when configured on the switch as the authentication failure action. If authentication succeeds, the client is authenticated, and the policies returned by the RADIUS server are applied. When both 802.1X authentication and MAC authentication methods are configured, authentication is performed depending on the authentication order, as explained in the following sections.

Authentication Order: 802.1X Authentication Followed by MAC Authentication

When the 802.1X authentication and MAC authentication methods are enabled on the same port, the default authentication order is to perform 802.1X authentication followed by MAC authentication (refer to Figure 6).

When 802.1X authentication succeeds, the client is authenticated, and the policies returned by the RADIUS server are applied. MAC authentication is not performed in this case. If 802.1X authentication fails and MAC authentication override is configured, MAC authentication is attempted; otherwise, the failure action is carried out. If the client does not respond to dot1x messages, MAC authentication is attempted after the client is declared non-dot1x-capable. On successful MAC authentication, the client is authenticated, and the policies returned by the RADIUS server are applied. On authentication failure, the configured failure action is applied.

FIGURE 6 Authentication Sequence: 802.1X Authentication Followed by MAC Authentication



Authentication Order: MAC Authentication Followed by 802.1X Authentication

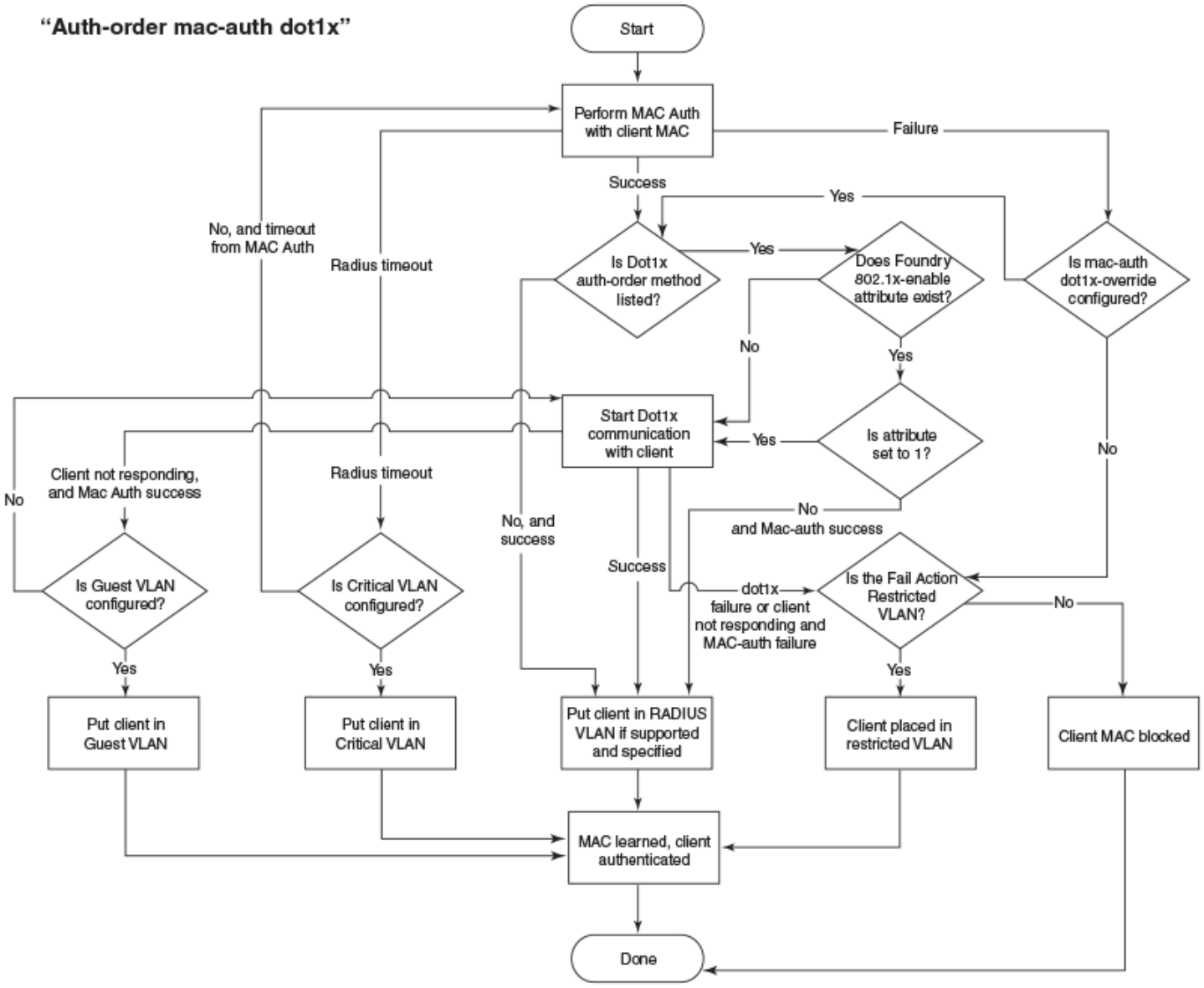
When the authentication order is set to perform MAC authentication followed by 802.1X authentication, by default, 802.1X authentication is performed, even if MAC authentication is successful (refer to Figure 7). On successful 802.1X authentication, the client is authenticated, and the policies returned by the RADIUS server are applied. On authentication failure, the configured failure action is applied.

The default behavior can be changed by specifying the RADIUS attribute (refer to Configuring Company-Specific Attributes on the RADIUS Server on page 41) to prevent the 802.1X authentication from being performed after successful MAC authentication or by configuring **mac-authentication dot1x-disable**. In this case, the client is authenticated, and the policies returned by the RADIUS server are applied after successful MAC authentication.

When MAC authentication fails, 802.1X authentication is not attempted, and the configured failure action is applied. However, if the **mac-authentication dot1x-override** command is configured, the clients that failed MAC authentication undergo 802.1X authentication. If 802.1X authentication is successful, the policies returned by the RADIUS server are applied to the port. If 802.1X authentication fails, the failure action is applied to the client.

When the timeout-action is "success" and the client is dot1x-capable, both authentication methods are tried. The client is placed in the auth-default VLAN, and EAP-SUCCESS is sent by the device. When the timeout-action is "critical-vlan" and the client is dot1x-capable, both authentication methods are tried, the client is placed in the critical VLAN, and EAP-SUCCESS is sent by the device.

FIGURE 7 Authentication Sequence: MAC Authentication Followed by 802.1X Authentication



The following list describes how Flexible authentication works in various success, failure, timeout, and dynamic VLAN assignment scenarios:

- When authentication succeeds and RADIUS returns VLAN information, the client is dynamically assigned to the RADIUS-assigned VLAN (the MAC address of the client is assigned to the VLAN), and authorization is carried out, depending on the attributes returned from the RADIUS server. For more information, refer to Dynamic VLAN Assignment on page 191.
- If RADIUS does not return any VLAN information after authentication, the client is placed in the auth-default VLAN.
- If the authentication fails, the failure action is carried out as per the configured failure action, for example, blocking the client or moving the client to the restricted VLAN.
- When the RADIUS server times out and authentication timeout action is configured as "success", the client is authenticated in the auth-default VLAN or the previously authenticated VLAN, depending on the following conditions:
  - If the RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the auth-default VLAN.

  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation.
- When the RADIUS server times out and authentication timeout action is configured as "critical-VLAN", the client is authenticated in the critical VLAN or the previously authenticated VLAN, depending on the following conditions:
  - If the RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the critical VLAN.
  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation.
- When the RADIUS server times out and the authentication timeout action is configured as "failure", the configured auth-failure-actions is performed, for example, moving the client to the restricted VLAN or blocking the client.
- During authentication, when RADIUS returns ACLs and the ACLs are not configured on the ICX device, the client authentication fails by default, resulting in the client being blocked.

Configuration Considerations and Guidelines for Flexible Authentication

- In Flexible authentication, ICX 8200 switches require a Layer 2 ingress ACL per user in multiple-untagged mode. The user will be blocked if the ACL filter is not installed due to hardware resources.
- In FastIron 08.0.90 and later releases, there will no longer be any link flap when a port is being added as a tagged member to a VLAN for the first time or when a port is being removed from the last tagged VLAN. External devices that may have relied on this link flap in the past for any kind of renegotiation must be reconfigured appropriately, or you must manually flap the interface.
- The RADIUS server must be configured to support a specific authentication method or a combination of authentication methods. A RADIUS server can be configured to use only 802.1X authentication, MAC authentication, Web authentication, or a combination of these authentication methods. However, from FastIron release 09.0.10b, any RADIUS server configured at the interface level will work for both 802.1x and Mac auth authentication methods, no matter what authentication methods are configured globally. For more information about RADIUS configuration, refer to RADIUS Authentication on page 37.
- Avoid configuring both secure and non-secure RADIUS servers in the same RUCKUS ICX network. For example, avoid using both UDP and TLS connections to RADIUS servers from an ICX switch configuration.
- MACsec and Flexible authentication cannot be configured on the same port.
- Flexible authentication and MVRP cannot be enabled on the same port.
- MVRP dynamic VLANs cannot be configured as authentication VLANs (auth-default VLAN, guest VLAN, critical VLAN, or restricted VLAN). This limitation applies at both the global configuration and interface configuration levels.
- Flexible authentication cannot be enabled on ports that have any of the following features enabled:
  - Link aggregation
  - Mirroring
  - Static IPv4/IPv6 ACL
  - Selective Q-in-Q
  - VLAN-mapping
  - Route-only
- Incoming traffic on unauthenticated ports is blocked by ICX devices, while allowing for outgoing broadcasts and multicasts to account for waking connected devices that are in a sleep state. This is the default behavior, and there is no configuration option.
- Once a VLAN is assigned as an authentication special VLAN (a critical VLAN, voice VLAN, guest VLAN, or restricted VLAN), it is not advisable to delete the VLAN. If the VLAN is deleted, you must take care of global authentication changes and impact.
- It is not advisable to assign a non-existent VLAN as an authentication special VLAN, particularly at the interface level.
- If Web authentication is enabled on the restricted VLAN, critical VLAN, guest VLAN, or a RADIUS-assigned VLAN, the device uses Web authentication as a fallback or additional authentication method. Web authentication can be enabled on eight VLANs.

- The client session establishes a relationship between the username and MAC address used for authentication. If attempting to gain access from different clients (with different MAC addresses), you must be authenticated from each client.
- When a client is denied access to the network, its session is aged out if no traffic is received from the client MAC address over a default hardware aging period (70 seconds), plus a software aging period. Both these age intervals can be changed, or session aging can be disabled altogether. After the denied client session is aged out, traffic from that client is no longer blocked, and the client can be reauthenticated.
- In the **authentication filter** command used to configure authentication override for specified MAC addresses, the statements **permit any any** and **deny any any** are not allowed.
- When dox1x port-control is configured as "force-authorized" or "force-unauthorized" on an interface, interface authentication configuration is not allowed on that interface.
- When Flex Auth is configured on a Multi-Chassis Trunking (MCT) Cluster Edge Port (CEP) that has a previously configured MCT VLAN, the dynamic VLAN returned for the user by RADIUS must be the same as the MCT VLAN.
- RADIUS attributes such as VLAN or ACL assignments cannot be changed dynamically in the RADIUS server when a Flexible authentication session is already authenticated. Likewise, a VLAN or ACL cannot be changed when an authenticated client reauthenticates. To apply an updated attribute, the related session must be cleared manually, and the attribute must be modified in the RADIUS server before the client is authenticated.

802.1X Authentication

ICX devices support the IEEE 802.1X standard for authenticating devices attached to LAN ports. Using 802.1X, you can configure an ICX device to grant access to a port based on information supplied by a client to an authentication server.

When a user logs in to a network that uses 802.1X, the ICX device grants (or does not grant) access to network services after the user is authenticated by an authentication server. The user-based authentication in 802.1X provides an alternative to granting network access based on a user IP address, MAC address, or subnetwork.

The ICX implementation of 802.1X supports the following RFCs:

- RFC 2284 PPP Extensible Authentication Protocol (EAP)
- RFC 2865 Remote Authentication Dial In User Service (RADIUS)
- RFC 2869 RADIUS Extensions

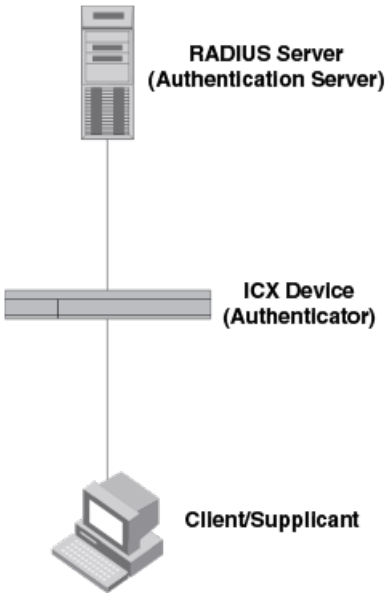
Device Roles in an 802.1X Configuration

The 802.1X standard defines the roles of client (supplicant), authenticator, and authentication server in a network.

The client (known as a supplicant in the 802.1X standard) provides username and password information to the authenticator. The authenticator sends this information to the authentication server. Based on the client's information, the authentication server determines whether the client can use services provided by the authenticator. The authentication server passes this information to the authenticator, which then provides services to the client, based on the authentication result.

The following figure illustrates these roles.

FIGURE 8 Authenticator, Client (Supplicant), and Authentication Server in an 802.1X Configuration



Authenticator: The device that controls access to the network. In an 802.1X configuration, the ICX device serves as the authenticator. The authenticator passes messages between the client and the authentication server. Based on the identity information supplied by the client and the authentication information supplied by the authentication server, the authenticator either grants or does not grant network access to the client.

Client/supplicant: The device that seeks to gain access to the network. Clients must be running software that supports the 802.1X standard (for example, the Windows 7 operating system). Clients can either be directly connected to a port on the authenticator or through a hub.

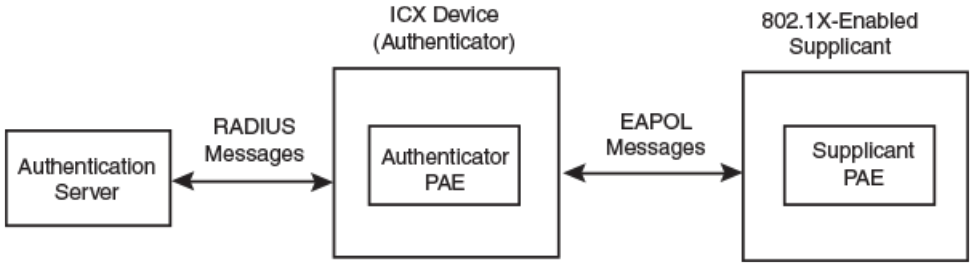
Authentication server: The device that validates the client and specifies whether or not the client may access services on the device. ICX devices support authentication servers running RADIUS.

Communication Between the Devices

For communication between the devices, 802.1X uses the Extensible Authentication Protocol (EAP), defined in RFC 2284. The 802.1X standard specifies a method for encapsulating EAP messages so that they can be carried over a LAN. This encapsulated form of EAP is known as EAP over LAN (EAPOL). The standard also specifies a means of transferring the EAPOL information between the client/supplicant, authenticator, and authentication server.

EAPOL messages are passed between the Port Access Entity (PAE) on the supplicant and the authenticator. The following figure shows the relationship between the authenticator PAE and the supplicant PAE.

FIGURE 9 Authenticator PAE and Supplicant PAE



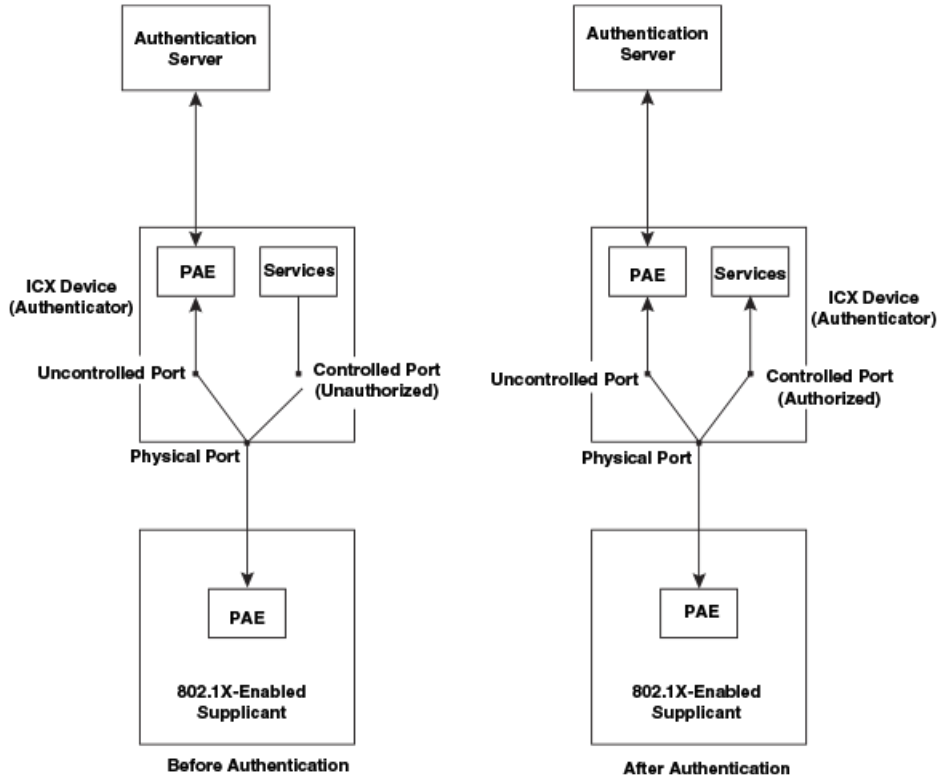
Authenticator PAE: The authenticator PAE communicates with the supplicant PAE, receiving identifying information from the supplicant. Acting as a RADIUS client, the authenticator PAE passes the supplicant information to the authentication server, which decides whether the supplicant can gain access to the port. If the supplicant passes authentication, the authenticator PAE grants it access to the port.

Supplicant PAE: The supplicant PAE supplies information about the client to the authenticator PAE and responds to requests from the authenticator PAE. The supplicant PAE can also initiate the authentication procedure with the authenticator PAE, as well as send log off messages.

Controlled and Uncontrolled Ports

A physical port on the device used with 802.1X authentication has two virtual access points: a controlled port and an uncontrolled port. The controlled port provides full access to the network. The uncontrolled port provides access only for EAPOL traffic between the client and the authenticator. When a client is successfully authenticated, the controlled port is opened to the client. The following figure illustrates this concept.

FIGURE 10 Controlled and Uncontrolled Ports Before and After Client Authentication



Before a client is authenticated, only the uncontrolled port on the authenticator is open. The uncontrolled port allows only EAPOL frames to be exchanged between the client and the authenticator. The controlled port is in the unauthorized state and allows no traffic to pass through.

During authentication, EAPOL messages are exchanged between the supplicant PAE and the authenticator PAE, and RADIUS messages are exchanged between the authenticator PAE and the authentication server. If the client is successfully authenticated, the controlled port becomes authorized for that client, and traffic from the client can flow through the port normally. When a client connected to the port is successfully authenticated, the client is authorized to send traffic through the controlled port until the client logs off.

Port Control for Authentication

To activate authentication on an 802.1X-enabled interface, you must specify the kind of port control to be used on the interface.

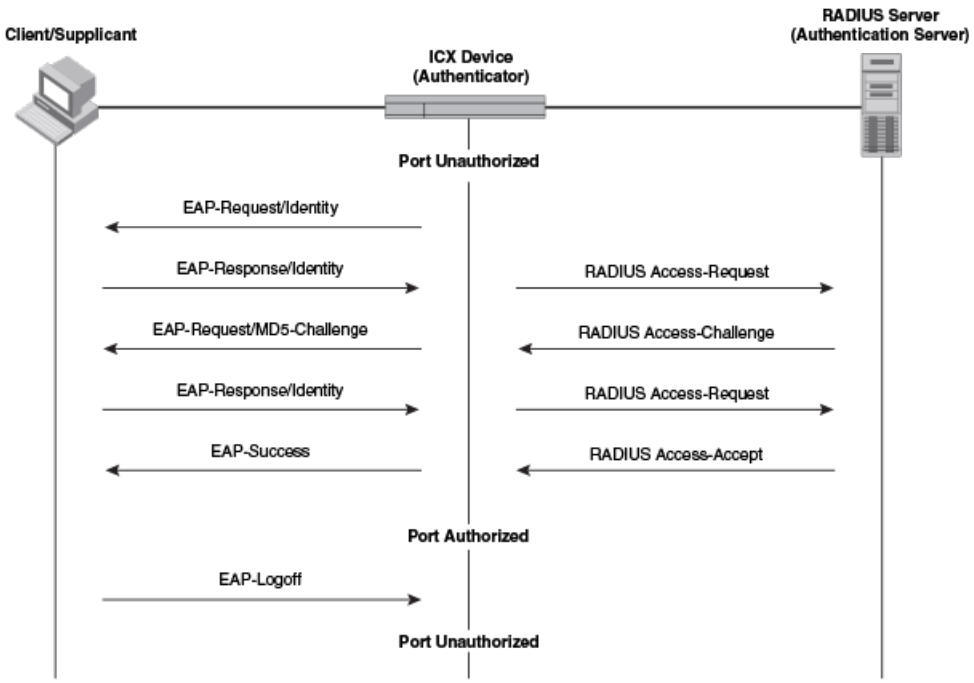
The port control can be one of the following types:

- Force-authorized: The controlled port is placed unconditionally in the authorized state, allowing all traffic. This is the default state for ports on the ICX device.
- Force-unauthorized: The controlled port is placed unconditionally in the unauthorized state.

Message Exchange During Authentication

The following figure illustrates a sample exchange of messages between an 802.1X-enabled client, an ICX device acting as the authenticator, and a RADIUS server acting as an authentication server.

FIGURE 11 Message Exchange between Client (Supplicant), Authenticator, and Authentication Server



In this example, the authenticator (the ICX device) initiates communication with an 802.1X-enabled client. When the client responds, it is prompted for a username (255 characters maximum) and password. The authenticator passes this information to the authentication server, which determines whether the client can access services provided by the authenticator. When the client is successfully authenticated by the RADIUS server, the port is authorized. When the client logs off, the port becomes unauthorized again.

The ICX 802.1X implementation supports dynamic VLAN assignment. If one of the attributes in the Access-Accept message sent by the RADIUS server specifies a VLAN identifier, the client port becomes a MAC-VLAN member of the specified VLAN. When the client disconnects from the network, the port is removed from the authorized VLAN.

The ICX 802.1X implementation supports dynamically applying an IP ACL to a port in either the ingress or egress direction, based on information received from the authentication server.

If a client does not support 802.1X, authentication cannot take place. The ICX device sends EAP-Request/Identity frames to the client, but the client does not respond to them. When a client that supports 802.1X attempts to gain access through a non-802.1X-enabled port, it sends an EAP start frame to the ICX device. When the ICX device does not respond, the client considers the port to be authorized and starts sending normal traffic.

ICX devices support identity and MD5-challenge requests in EAP Request/Response messages, as well as the following 802.1X authentication challenge types:

- EAP-TLS (RFC 2716): EAP Transport Level Security (TLS) provides strong security by requiring both the client and the authentication server to be identified and validated through the use of public key infrastructure (PKI) digital certificates. EAP-TLS establishes a tunnel between the client and the authentication server to protect messages from unauthorized user eavesdropping activities. Because EAP-TLS requires PKI digital certificates on both the clients and the authentication servers, the roll out, maintenance, and scalability of this authentication method is much more complex than other methods. EAP-TLS is best for installations with existing PKI certificate infrastructures.
- EAP-TTLS (Internet-Draft): The EAP Tunnelled Transport Level Security (TTLS) is an extension of EAP-TLS. Like TLS, EAP-TTLS provides strong authentication measures; however, it requires only the authentication server to be validated by the client through a certificate exchange between the server and the client. Clients are authenticated by the authentication server using usernames and passwords.

  A TLS tunnel can be used to protect EAP messages and existing user credential services such as Active Directory, RADIUS, and LDAP. Backward compatibility for other authentication protocols such as PAP, CHAP, MS-CHAP, and MS-CHAP-V2 are also provided by EAP-TTLS. EAP-TTLS is not considered foolproof and can be fooled into sending identity credentials if TLS tunnels are not used. EAP-TTLS is suited for installations that require strong authentication measures without the use of mutual PKI digital certificates.
- PEAP (Internet-Draft): Protected EAP (PEAP) is an Internet-Draft that is similar to EAP-TTLS. A PEAP client authenticates directly with the back-end authentication server. The authenticator acts as a pass-through device that does not need to understand the specific EAP authentication protocols.

  Unlike EAP-TTLS, PEAP does not natively support usernames and passwords to authenticate clients against an existing user database such as LDAP. PEAP secures the transmission between the client and the authentication server with a TLS-encrypted tunnel. PEAP also allows other EAP authentication protocols to be used. It relies on the mature TLS keying method for its key creation and exchange. PEAP is best suited for installations that require strong authentication without the use of mutual certificates.

Configuration for these challenge types is the same as for the EAP-MD5 challenge type.

NOTE
If the 802.1X client sends a packet larger than 1500 bytes, you must use the **jumbo** command at the global configuration level of the CLI. If the supplicant or the RADIUS server does not support jumbo frames and if jumbo support is enabled on the switch, you can set the CPU IP MTU size. For more information on setting IP MTU size, refer to the "IP Addressing" chapter in the *RUCKUS FastIron Layer 3 Routing Configuration Guide*.

EAP Pass-Through Support

EAP pass-through is supported on ICX devices that have 802.1X enabled. EAP pass-through support is fully compliant with RFC 3748, in which, by default, compliant pass-through authenticator implementations forward EAP challenge request packets of any type.

MAC Authentication

MAC authentication is a way to configure an ICX device to forward or block traffic from a client (MAC address) based on authentication information received from a RADIUS server.

MAC authentication is a mechanism by which the ICX device switches or forwards incoming traffic originating from a specific MAC address only if a RADIUS server successfully authenticates the source MAC address. The MAC address itself is used as the username and password for RADIUS authentication. The user does not need to provide a specific username and password to gain access to the network.

The ICX implementation supports dynamic VLAN assignment. If one of the attributes in the Access-Accept message sent by the RADIUS server specifies a VLAN identifier, the client port becomes a MAC-VLAN member of the specified VLAN. When the client disconnects from the network, the port is removed from the authorized VLAN.

The ICX implementation also supports dynamically applying an IP ACL to a port in the ingress or egress direction, based on information received from the authentication server.

If the RADIUS server cannot validate the user's MAC address, it is considered an authentication failure, and a specified authentication failure action is applied. The default authentication failure action is to drop traffic from the non-authenticated MAC address in hardware. You can also configure the device to move the port on which the non-authenticated MAC address was learned into a restricted VLAN.

MAC Address Formats Sent to the RADIUS Server

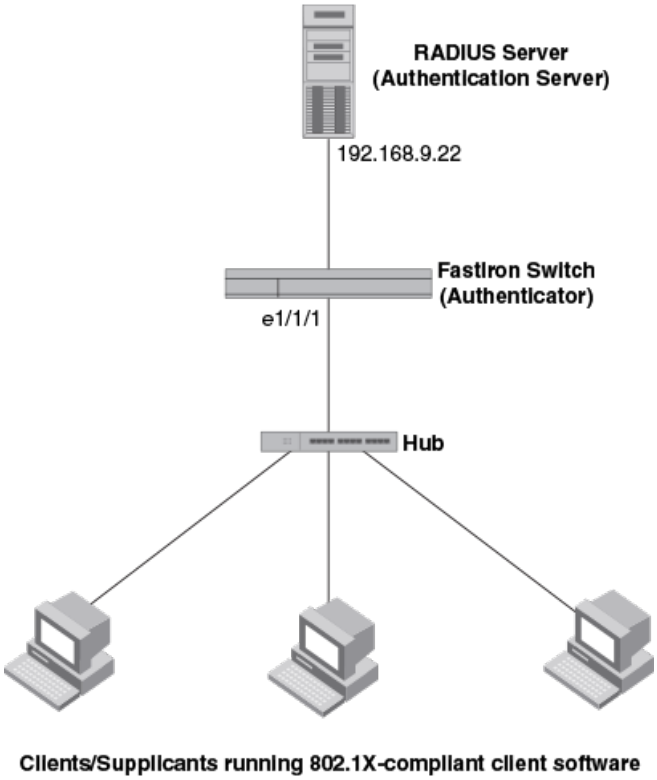
The MAC address of the client is used as the username and password for authentication.

When MAC authentication is configured, the ICX device authenticates the client using the MAC address and the RADIUS server. The device uses the MAC address for both the username and the password in the request sent to the RADIUS server. Several formats can be used to send the MAC address to the RADIUS server, including sending the MAC address in uppercase. The format is configurable using the **mac-authentication password-format** command. The lowercase option and xxxxxxxxxxxx are the default format.

Authenticating Multiple Hosts Connected to the Same Port

ICX devices support 802.1X authentication for ports with more than one host connected to them. The following figure illustrates a sample configuration where multiple hosts are connected to a single port.

FIGURE 12 Multiple Hosts Connected to a Single 802.1X-enabled Port



If multiple hosts are connected to a single 802.1X or MAC authentication-enabled port, the ICX device authenticates each of them individually. Each host authentication status is independent of the others, so that if one authenticated host disconnects from the network, it has no effect on the authentication status of any of the other authenticated hosts.

By default, traffic from hosts that cannot be authenticated by the RADIUS server is dropped in hardware. As an option, you can configure the ICX device to assign the port to a "restricted" VLAN if authentication of the client is unsuccessful.

How Flexible Authentication Works for Multiple Clients

When multiple hosts are connected to a port on an ICX device, Flexible authentication is performed in the following way.

1. One of the 802.1X-enabled clients attempts to log in to a network in which an ICX device serves as an authenticator.
2. The ICX device creates an internal session for the client. The session serves to associate a client MAC address and username with its authentication status. Users trying to gain access from different clients (with different MAC addresses) must be authenticated from each client.
3. The ICX device performs 802.1X authentication or MAC authentication for the client. Messages are exchanged between the ICX device and the client and between the ICX device and the authentication (RADIUS) server. The result of this process is that the client is either successfully authenticated or not authenticated, based on the access policy configured on the RADIUS server.
4. If the client is successfully authenticated, the client session is set to "access-allowed." This means that traffic from the client can be forwarded normally.
5. If authentication for the client is unsuccessful, an authentication failure action is applied. The authentication failure action can be either to drop traffic from the client or to place the client in a restricted VLAN:
   - If the authentication failure action is to drop traffic from the client, the client session is set to "access-denied." This causes traffic from the client to be dropped in hardware.
   - If the authentication failure action is to place the port in a restricted VLAN, the client session is set to "access-restrict." The port is moved to the specified restricted VLAN, and traffic from the client is forwarded normally.
6. The previous steps are repeated for all clients.
7. When the client disconnects from the network, the ICX device deletes the client session. This does not affect the session or authentication status (if any) of the other hosts connected on the port.
8. If any of the clients have IP ACLs sent by the RADIUS server, all such user ACLs are applied to the respective client sessions.

Flexible Authentication Accounting

When 802.1X authentication or MAC authentication is enabled on the ICX device, RADIUS accounting can also be enabled. This feature enables the ICX device to log information on the RADIUS server about 802.1X-authenticated clients or MAC-authenticated clients. The information logged on the RADIUS server includes the client session ID, MAC address, user name, authenticating physical port, packet statistics, client IP address information, and so on. Flexible authentication accounting works in the following way.

1. A RADIUS server successfully authenticates a client.
2. If 802.1X accounting or MAC authentication accounting is enabled, the ICX device sends an 802.1X Accounting Start packet to the RADIUS server, indicating the start of a new session.
3. The RADIUS server acknowledges the Accounting Start packet.
4. The RADIUS server records information about the client.
5. Periodically or whenever a change occurs in the session, such as an IP address update, the ICX device sends an Interim Accounting Stop packet to the RADIUS server.
6. The RADIUS server acknowledges the Interim Accounting Stop packet.
7. When the session is concluded, the ICX device sends an Accounting Stop packet to the RADIUS server, indicating the end of the session.

8. The RADIUS server acknowledges the Accounting Stop packet.

For more information, refer to RADIUS Accounting for 802.1X Authentication and MAC Authentication on page 55.

Change of Authorization

Change of authorization (CoA) allows administrators to change authorization dynamically after the device or user is authenticated. As part of authorization, the user or device is given access to specific resources on the network based on the policies or commands downloaded from the RADIUS server. The CoA allows the administrator to change these policies without terminating the sessions. A CoA request packet can be sent by the CoA client (typically a RADIUS or policy server) to change the session authorizations on the ICX device. The request identifies the device and the sessions to be authorized. The following table explains the CoA commands and relevant attributes needed.

NOTE
Only relevant attributes must be present in the CoA request packet. Any additional attributes present in the CoA Request packet sent from the RADIUS client may cause the CoA request to fail (may result in a NAK).

TABLE 20 CoA Commands and RADIUS Attributes

CoA Command	Description	RADIUS Attributes
Disconnect	Disconnect the specified session	Calling-Station-Id(31), NAS-IP-Address(4)
Modify ACL	Modify ingress/egress ACLs on specified session	Calling-Station-Id(31), NAS-IP-Address(4), Filter-Id(11)
Disable Port	Disable the specified port	Calling-Station-Id(31), NAS-IP-Address(4), Ruckus FlexAuth AVP(20)
Flip Port	Flip the specified port (disable and enable)	Calling-Station-Id(31), NAS-IP-Address(4), Ruckus FlexAuth AVP(20)
Reauth Host	Reauthenticate the specified host (session)	Calling-Station-Id(31), NAS-IP-Address(4), Ruckus FlexAuth AVP(20)

For more information on attributes for RADIUS to support CoA, refer to Configuring Company-Specific Attributes on the RADIUS Server on page 41.

Multiple RADIUS Servers

Flexible authentication communicates with the RADIUS server to authenticate a new client or reauthenticate an already authenticated client. The ICX device supports multiple RADIUS servers. If communication with one of the RADIUS servers times out, the others are tried in sequential order. If a response from a RADIUS server is not received within a specified time (by default, 3 seconds), the RADIUS request times out, and the device retries the request up to three times. If no response is received, that RADIUS server is marked as down, and the next available RADIUS server is chosen, until all servers are exhausted, or a response is received.

Marking the RADIUS server as down helps in making the authentication process faster, as only the available servers are contacted. When configured, the servers that are down are periodically contacted to check if they are available, and when they become available, they are marked accordingly.

There are several professional and experimental quality RADIUS servers, and all servers are configured with the usernames and passwords of authenticated users. For MAC authentication, the username and password are the MAC address itself. The ICX device uses the MAC address for both the username and the password in the request sent to the RADIUS server. For 8021.X authentication, the username and password are typically configured as unique IDs, which the clients use when they log in to the network. For example, with a MAC address of 00:10:94:00:fe:aa, the user's file on the RADIUS server is configured with the username and password both set to 00:10:94:00:fe:aa. If a user using dot1x authentication has to authenticate from the same device, the user profile may include name and password.

Session Aging

Session aging is enabled by default, and all sessions are monitored for inactivity so that they can be purged when they are inactive for longer periods of time to conserve resources.

Aging for Permitted MAC Addresses

Aging for a permitted or non-blocked MAC address occurs in two phases, known as MAC aging and software aging.

The MAC aging time for non-blocked MAC addresses is the length of time specified with the **mac-age-time** command. The software aging period for non-blocked MAC addresses is configurable, using the **max-sw-age** command (the default is 120 seconds). When the MAC aging period ends, the software aging period begins. When the software aging period ends, the session is aged out.

Aging for Denied MAC Addresses

Software aging is not applicable for blocked MAC addresses. The hardware aging period for blocked MAC addresses is set to 70 seconds by default, and it can be configured using the **max-hw-age** command. Once the hardware aging period ends, the blocked MAC address ages out so that the session is deleted and can be authenticated again if the ICX device receives traffic from the MAC address.

Disabling MAC Address Aging

Aging can be disabled for all MAC sessions globally or at the interface level to prevent the MAC sessions from being aged out.

You can disable aging of either the permitted (authenticated and restricted) sessions or the denied sessions. If **disable-aging** is configured for permitted MAC sessions, only the permitted sessions are prevented from being aged out, while the denied sessions age out after the hardware aging period. If **disable-aging** is configured for denied sessions, only the denied sessions are prevented from being aged out, while the permitted sessions age out based on the MAC aging interval configured using the **mac-age-time** command plus the software aging period.

Periodic Reauthentication of Authenticated Clients

NOTE
Reauthentication is enabled by default for restricted VLANs and critical VLANs. Reauthentication is not supported for guest VLANs.

ICX devices can be configured to periodically reauthenticate the authenticated clients that are connected to interfaces enabled for 802.1X authentication and MAC authentication. When periodic reauthentication is enabled using the **re-authentication** command, the ICX device reauthenticates clients every 3,600 seconds (1 hour) by default. The reauthentication interval is configurable using the **reauth-period** command. The RADIUS server can overwrite this interval for each client using the Session-Timeout and Termination-Action attributes.

NOTE
With the dead RADIUS server enhancement, RADIUS servers are monitored and marked dead when they do not respond. When no RADIUS servers are available, a client reauthentication attempt results in a timeout. Therefore, when no servers are available, reauthentication is not performed. Instead, the timeout action is performed, and the client continues with the same credentials when the **auth-timeout-action** command configuration is set to success or critical-vlan. This avoids potential issues with some 802.1X clients that are known to go into an inconsistent state after client reauthentication times out and when the **auth-timeout-action** command configuration is set to success.

Denial of Service Protection Support

A Denial of Service (DoS) attack can occur against the ICX device when a high volume of new source MAC addresses is sent to the device, causing the CPU to be overwhelmed with performing RADIUS authentication for these MAC addresses. In addition, the high CPU usage in such an attack could prevent the RADIUS response from reaching the CPU in time, causing the device to make additional authentication attempts.

You can enable Denial of Service protection using the **authentication dos-protection** command in interface configuration mode. The ICX device does not start forwarding traffic from an authenticated MAC address in hardware until the RADIUS server authenticates the MAC address. Traffic from the non-authenticated MAC addresses is sent to the CPU.

SNMP Traps for Flexible Authentication

You can enable and disable SNMP traps for Flexible authentication using the **snmp-server enable traps mac-authentication** command. SNMP traps are enabled by default.

Syslog Messages for Flexible Authentication

Syslog messages are always enabled and sent to the configured log server and to internal logging for significant events during the authentication of clients. Events such as user logged-in, user logged-off, and RADIUS accepted or rejected with a list of attributes are logged.

RADIUS Attributes for Authentication and Accounting

Various RADIUS attributes are supported for 802.1X authentication and MAC authentication and accounting.

RADIUS attributes are used to define specific authentication, authorization, and accounting (AAA) elements in a user profile, which is stored in the RADIUS server. When a client successfully completes the EAP authentication process, the authentication (RADIUS) server sends the authenticator (the ICX device) a RADIUS Access-Accept message that grants the client access to the network. The Access-Accept message contains attributes set for the user in the user's access profile on the RADIUS server. The user's access profile is used for many functions, such as dynamic VLAN assignment, dynamic IP ACL assignment, session timeout, and authentication order rules for Flexible authentication.

The following table describes the RADIUS attributes that are supported for 802.1X authentication and MAC authentication.

TABLE 21 RADIUS Attributes for 802.1X Authentication and MAC Authentication

Attribute Name	Attribute ID	Data Type	Description
Acct-Interim-Interval	85	integer	Indicates the interval for sending the interim updates to the RADIUS server in seconds. The minimum value is 300 seconds.
Calling-Station-Id	31	string	The supplicant MAC address in ASCII format (uppercase only), with octet values separated by a dash (-). For example, 00-00-00-23-19-C0.
Class	25	string	Sent by the server to the client in an Access-Accept. This attribute should not be modified by the client; it should be sent to the accounting server when accounting is supported.
Idle-Timeout	28	integer	Idle timeout after which the session is cleared when there is no traffic. This is equivalent to configuring the **max-sw-age** command through the CLI.

TABLE 21 RADIUS Attributes for 802.1X Authentication and MAC Authentication (continued)

Attribute Name	Attribute ID	Data Type	Description
NAS-Identifier	32	string	Network access server identifier (the hostname of the device).
NAS-IP-Address	4	integer	IP address of the network access server requesting user authentication.
NAS-Port	5	integer	Physical network access server port number that is authenticating the user.
NAS-Port-Id	87	string	Identifier of the network access server port that is authenticating the user.
NAS-Port-Type	61	integer	The port type (physical or virtual) that is authenticating the user.
Service-Type	6	integer	The type of service that is requested by the user or to be provided to the user.
Session-Timeout	27	integer	The maximum number of seconds of service provided to the user before termination of the session or prompt.
Termination-Action	29	integer	The action to be taken by the network access server when the specified service is completed.
Tunnel-Private-Group-ID	81	string	Group identifier for a particular tunneled session.
Tunnel-Medium-Type	65	integer	Indicates which transport medium to use when creating a tunnel for protocols that can operate over multiple transport mediums.
Tunnel-Type	64	integer	Indicates the tunnel protocol that is either in use by a tunnel terminator or to be used by a tunnel initiator.
User-Name	1	string	Indicates the name of the user to be authenticated.

NOTE

Any of the default or configured values on the ICX device are replaced with RADIUS sent attributes because RADIUS values always take precedence over the values configured on the ICX device.

The following table describes the RADIUS attributes that are supported for 802.1X accounting and MAC accounting.

TABLE 22 RADIUS Attributes for 802.1X Accounting and MAC Accounting

Attribute Name	Attribute ID	Data Type	Description
Acct-Authentic	45	Integer	Indicates how the user was authenticated: 1: RADIUS 2: Network access server itself 3: Other remote authentication protocol
Acct-Delay-Time	41	Integer	Number of seconds that the client has been trying to send this record.

TABLE 22 RADIUS Attributes for 802.1X Accounting and MAC Accounting (continued)

Attribute Name	Attribute ID	Data Type	Description
Acct-Input-Octets	42	Integer	Number of octets received from the port while this service is being provided. This attribute can only be present in Accounting-Request records when the Acct-Status-Type is set to stop.
Acct-Input-Packets	47	Integer	Number of packets received from the port while this service is being provided to a framed user. This attribute can only be present in Accounting-Request records when the Acct-Status-Type is set to stop.
Acct-Output-Octets	43	Integer	Number of octets sent to the port while this service is being provided. This attribute can only be present in Accounting-Request records when the Acct-Status-Type is set to stop.
Acct-Output-Packets	48	Integer	Number of packets sent to the port while this service is being provided to a framed user. This attribute can only be present in Accounting- Request records when the Acct- Status-Type is set to stop.
Acct-Session-Id	44	Integer	The account session ID, which is a number from 1 through 4294967295.
Acct-Session-Time	46	Integer	Number of seconds that the user has received service. This attribute can only be present in Accounting-Request records when the Acct-Status-Type is set to stop.
Acct-Status-Type	40	Integer	Indicates whether this Accounting Request marks the beginning (start) or end (stop) of the user service. It also indicates when to send interim updates to the RADIUS server on the status of an active session such as IP address change. The interim update includes the duration of the current session and information on current data usage. 1: Start 2: Stop 3: Interim Update
Acct-Terminate-Cause	49	Integer	Specifies the reason for session termination; for example, session timeout, idle timeout, user logoff, admin forced, port down or disabled, system reload, and so on. This attribute is sent out in an Accounting Stop request.
Framed-IPv4-Address	8	IPv4 address	IPv4 address that is assigned on the host or IP routing residential gateway to the interface facing the network access server.

TABLE 22 RADIUS Attributes for 802.1X Accounting and MAC Accounting (continued)

Attribute Name	Attribute ID	Data Type	Description
Framed-IPv6-Address	168	IPv6 address	IPv6 address that is assigned on the host or IP routing residential gateway to the interface facing the network access server.
Framed-MTU	12	Integer	Indicates the maximum transmission unit (MTU) to be configured for the user.

Configuring ICX Vendor-Specific Attributes on the RADIUS Server

If the RADIUS authentication is successful, the RADIUS server sends an Access-Accept message to the ICX device, authenticating the client. The Access-Accept message can include attributes that specify additional information about the client. If MAC authentication and 802.1X authentication are configured on the same port, attributes listed in the following tables can be configured on the RADIUS server.

Attributes should be added to the RADIUS server configuration and configured in the individual or group profiles of the devices to be authenticated. The ICX device supports two Vendor IDs: 1991 (Foundry) and 25053 (RUCKUS). The Foundry Vendor ID is supported for backward compatibility. For more information, refer to Configuring RADIUS (Overview) on page 40. The following tables list the attributes for Foundry and RUCKUS VSAs.

NOTE
These vendor-specific attributes are optional.

TABLE 23 Foundry VSAs for RADIUS

Attribute Name	Attribute ID	Data Type	Description
Foundry-802_1x-enable	6	integer	Specifies whether 802.1X authentication is performed when MAC authentication is successful for a device. This attribute can be set to one of the following values: 0: Do not perform 802.1X authentication on a device that passes MAC authentication. 1: Perform 802.1X authentication when a device passes MAC authentication.

TABLE 23 Foundry VSAs for RADIUS (continued)

Attribute Name	Attribute ID	Data Type	Description
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 values: 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.
Foundry-COA-Command-List	10	string	Specifies the CoA command to be performed. This attribute can be set to one of the following values: Reauth-host: Reauthenticate the host. Disable-port: Disable the port. Flip-port: Flip or reset the port.
Foundry-Voice-Phone-Config	11	string	Identifies the client as a voice phone device and optionally specifies the voice phone device configuration. When the device is a phone, LLDP/CDP is configured to "voice VLAN to phone" so the phone uses a voice session. LLDP supports additional options, such as differentiated services code point (DSCP) and priority, to configure MED policy. The following options can be specified through this VSA: “ “: DSCP:46, priority 5 (default) “dscp:40; priority:4”: DSCP: 40, priority: 4 “dscp:30”: DSCP: 30, priority: 5 “priority:7”: DSCP: 46, priority: 7
Foundry-NAS-Ipv4-Filter-Rule	12	string	Includes an IPv4 ACL filter statement that is received and programmed on the ICX device for the duration of the RADIUS session.
Foundry-NAS-Ipv6-Filter-Rule	13	string	Includes an IPv6 ACL filter statement that is received and programmed on the ICX device for the duration of the RADIUS session.

TABLE 24 RUCKUS VSAs for RADIUS for Flexible Authentication

Attribute Name	Attribute ID	Data Type	Description
RUCKUS FlexAuth AVP	20	string	The generic name of the attribute is value-pair attribute, which can specify the following attributes (similar to the Foundry VSAs): dot1x-enable (same as Foundry-802.1x-enable) dot1x-valid (same as Foundry-802.1x-valid) coa-attr (same as Foundry-COA-Command-List) voice-phone (same as Foundry-Voice-Phone-Config)

Support for the RADIUS User-Name Attribute in Access-Accept Messages

ICX devices support the RADIUS user-name (type 1) attribute in the Access-Accept message returned during authentication.

In 802.1X authentication, the user-name attribute is useful when the client does not provide a username in the EAP-response/identity frame and the username is key to providing useful information.

In MAC authentication, the user-name attribute is useful to bind the username with the client MAC address because the client never provides it, and the username is key to providing useful information.

When sFlow forwarding is enabled on a Flexible authentication-enabled interface, the samples taken from the interface include the username string at the inbound or outbound port, or both, if that information is available. For more information on sFlow, refer to the *RUCKUS FastIron Monitoring Configuration Guide*.

For example, when the user-name attribute is sent in the Access-Accept message, it is then available for display in sFlow sample messages sent to a collector and in the output of some **show authentication** commands, such as **show authentication sessions** and **show authentication session detail**.

This same information is included as the user-name attribute of RADIUS accounting messages sent to RADIUS accounting servers.

TABLE 25 RADIUS User-Name Attribute Details

Attribute Name	Type	Value
user-name	1	*name* (string)

Dynamic VLAN Assignment

After authentication succeeds, a VLAN assignment policy can be applied to control the destination VLAN of the client. Dynamic VLAN assignment allows clients to connect to the network anywhere. Based on their credentials, clients are placed in the appropriate VLAN irrespective of the ports to which they are connected.

MAC authentication and 802.1X authentication support dynamic VLAN assignment, where a port can be placed in one or more VLANs based on the VLAN attribute sent from the RADIUS server. With dynamic VLAN assignment, the port becomes a MAC-VLAN member of the specified VLAN.

When a client is successfully authenticated, the RADIUS server sends a RADIUS Access-Accept message to the ICX device that allows the device to forward traffic from that client (using the MAC address of the client), including the VLAN information using RADIUS attributes to place the client in the designated VLAN. Refer to Configuring the RADIUS Server to Support Dynamic VLAN Assignment for Authentication on page 192 for a list of the attributes that must be set on the RADIUS server.

A port can be configured with one or multiple authentication methods. If only one authentication is performed, then the VLAN returned from that authentication is used. With multiple methods and based on authentication order, the VLAN from the last authentication is used. If the last authentication does not return any VLAN, the auth-default VLAN is used. This ensures that the user is always placed in a VLAN.

NOTE
A RADIUS-assigned VLAN can also serve as one or more of the following VLANs: restricted VLAN, critical VLAN, or guest VLAN.

Configuring the RADIUS Server to Support Dynamic VLAN Assignment for Authentication

Dynamic VLAN assignments from the RADIUS server can be enabled in multiple formats. VLAN assignments can be tagged, untagged, single, multiple, or a combination of tagged and untagged VLANs for different use cases, for example, with client devices such as computers, IP phones, wireless access points, or servers running hypervisors that have multiple Virtual Machines (VMs).

To specify VLAN identifiers on the RADIUS server, add the attributes in the following table to the device (client) profile for MAC authentication. For 802.1X authentication, add these attributes to the user (client) profile.

TABLE 26 Attributes for Dynamic VLAN Assignment

Attribute Name	Type	Value
Tunnel-Type	064	13 (decimal) - VLAN
Tunnel-Medium-Type	065	6 (decimal) - IEEE-802
Tunnel-Private-Group-ID	081	vlan-id or vlan-name (untagged) U:vlan-id of U:vlan-name (untagged) T:vlan-id or T:vlan-name (tagged)

NOTE
A maximum of 32 dynamic VLANs can be sent from RADIUS with a Tunnel-Private-Group-ID in the format U: multiple T: where U and T both represent either a VLAN-name or a VLAN-ID.

NOTE
Different formats are supported for Tunnel-Private-Group-ID in different FastIron releases. FastIron 08.0.30b and later releases support the following formats:

- vlan-id or vlan-name
- U:vlan-id or U:vlan-name
- T:vlan-id or T:vlan-name
- U:vlan-id T:vlan-name
- multiple T:<>
- U: multiple T:<>

The ICX device interprets the attributes as follows:

- If the Tunnel-Type or the Tunnel-Medium-Type attributes in the Access-Accept message do not have the specified values, the ICX device ignores these Attribute-Value pairs. If the Tunnel-Private-Group-ID is valid, the client is authorized in this VLAN; otherwise, it is authorized in the auth-default VLAN.

- When the ICX device receives and parses the Tunnel-Private-Group-ID attribute, it checks whether the vlan-name string matches the name of a configured VLAN or the vlan-id on the ICX device. If there is a VLAN match, the client port is placed in the VLAN.
- If the vlan-name string does not match either the name or the ID of a VLAN configured on the ICX device, the VLAN name or ID is created and then used.

Authentication Success Scenarios

The dynamic VLAN assignment depends on the various VLAN formats returned by the RADIUS server.

RADIUS Returns Only a VLAN Identifier or an Untagged VLAN Identifier

When the Access-Accept message returned by RADIUS contains the VLAN information in either vlan-id or vlan-name or U:vlan-id or U:vlan-name format:

- In single untagged mode (the default), the port membership is removed from the auth-default VLAN and added to the RADIUS-specified VLAN as a MAC-VLAN member.

  The following behavior is the default behavior of the device:

  - Subsequent clients that are authenticated with different dynamic VLANs are blocked.
  - If another client is authenticated on the same VLAN, it is permitted in the first client's dynamic VLAN.
  - If another client is authenticated on the port without a RADIUS VLAN, it is permitted in the first client's dynamic VLAN.
  - Once all the clients in the new VLAN age out, the port is moved back to the auth-default VLAN.
- In multiple untagged mode, the port is added as a MAC-VLAN member to the RADIUS-specified VLAN without removing its membership from the auth-default VLAN.

RADIUS Returns a Single or Multiple Tagged VLAN Identifiers

When the Access-Accept message returned by RADIUS contains the VLAN information in either T:vlan-id or T:vlan-id1; T:vlan-id2 format, for MAC authentication, if the authentication is triggered by a tagged packet and if the VLAN matches the tagged VLAN or VLAN list returned by RADIUS, the session is authenticated, and the port becomes a tagged member of all the dynamically assigned VLANs.

RADIUS Returns Untagged and Single or Multiple Tagged VLAN Identifiers

When the Access-Accept message returned by RADIUS contains the VLAN information in U:vlan-id1; T:vlan-id2 or U:vlan-id1; T:vlan-id2; T:vlan-id3; T:vlan-id4 format:

- In single untagged mode (the default), the port membership is removed from the auth-default VLAN and added to the RADIUS-specified VLAN as a MAC-VLAN member. It is also added to the tagged VLANs as a tagged member.

  The following behavior is the default behavior of the device:

  - Subsequent clients that are authenticated with different dynamic VLANs are blocked.
  - If another client is authenticated on the same VLAN, it is permitted in the first client's dynamic VLAN.
  - If another client is authenticated on the port without a RADIUS VLAN, it is permitted in the first client's dynamic VLAN.
  - Once all the clients in the new VLAN age out, the port is moved back to the auth-default VLAN.
- In multiple untagged mode, the port is added as a MAC-VLAN member to the RADIUS-specified untagged VLAN without removing its membership from the auth-default VLAN. It is also added to the tagged VLANs as a tagged member.

For MAC authentication, if the authentication is triggered by a tagged packet and if the VLAN matches the tagged VLAN or VLAN list returned by RADIUS, the session is authenticated, and the port becomes an untagged member of one VLAN and a tagged member of the other dynamically assigned VLANs.

NOTE
In single untagged mode (the default), if an authenticated client exists on a port and the second client trying to authenticate fails, the port is not moved to a restricted VLAN; instead, the second client is blocked.

Authentication Failure Scenarios

VLAN assignment in authentication failure cases depends on the authentication mode of the port being single untagged mode (the default) or multiple untagged mode. The authentication failure action configured applies to MAC authentication and 802.1X authentication and can be one of the following actions:

- Block the client's MAC address.
- Move the client to a restricted VLAN.

A failure action of moving the client to a restricted VLAN works only if the RADIUS server sends an Access-Reject message. Any other failure, for example, an undefined ACL or a VLAN movement error, results in blocking the client's MAC address.

A default ACL with IPv4 or IPv6 filters can also be configured to apply, if the failure action is a restricted VLAN.

NOTE
For the default ACL to work, a valid and defined ACL must be configured. Deleting the default ACL may result in unpredictable behavior. Dynamic modification of a default ACL is not supported. If changes must be made to the default ACL, clear all the existing sessions after the modifications are complete.

NOTE
For the dynamic ACL to work, a valid and defined ACL must be configured; otherwise, the session will be blocked. Deleting the dynamic ACL or any filter may result in unpredictable behavior. Dynamic modification to a dynamic ACL is not supported. If changes must be made to the dynamic ACL, clear all the applicable sessions after the modifications are complete. These actions should not affect other sessions or features.

An authentication failure action of blocking blocks the clients in both single and multiple untagged mode. If the authentication failure action is to move to a restricted VLAN, the behavior depends on whether the authentication mode is single untagged mode or multiple untagged mode.

In single untagged mode, the following behaviors apply:

- If the first client's authentication fails, the port membership is moved from the auth-default VLAN to the restricted VLAN.
- If other clients were authenticated previously on the same port, the new client is always blocked. Even after all other clients age out, the new client remains in the VLAN reserved for blocked clients until it ages out.
- If the previous sessions are in a restricted VLAN, the new client is moved to the restricted VLAN.
- If the previous sessions are in the critical VLAN or guest VLAN, the new client is blocked.
- In MAC authentication, if the authentication is initiated by a tagged packet, the client is blocked in the tagged VLAN irrespective of the configured failure action.

In multiple untagged mode, the following behaviors apply:

- If the failure action is configured as a restricted VLAN, the client is moved to the restricted VLAN. If the port is not part of the restricted VLAN, the port is made a MAC-VLAN member of the restricted VLAN.
- For MAC authentication, if the authentication is initiated by a tagged packet, the client is blocked in the tagged VLAN irrespective of the configured failure action.

Authentication Server Timeout Scenarios

VLAN assignment for RADIUS timeout cases depends on whether the authentication mode of the port is single untagged mode (the default) or multiple untagged mode. The authentication timeout action configured applies to MAC authentication and 802.1X authentication and can be one of the following actions:

- Retry
- Failure
- Success
- Move the client to a critical VLAN

A default ACL with IPv4 or IPv6 filters can also be configured to apply, if the timeout action is critical VLAN or Success.

NOTE
For the default ACL to work, a valid and defined ACL must be configured. Deleting the default ACL may result in unpredictable behavior. Dynamic modification of a default ACL is not supported. If changes must be made to the default ACL, clear all the existing sessions after the modifications are complete.

NOTE
For the dynamic ACL to work, a valid and defined ACL must be configured; otherwise, the session will be blocked. Deleting the dynamic ACL or any filter may result in unpredictable behavior. Dynamic modification to a dynamic ACL is not supported. If changes must be made to the dynamic ACL, clear all the applicable sessions after the modifications are complete. These actions should not affect other sessions or features.

Authentication timeout action depends on whether the authentication mode is single untagged mode or multiple untagged mode.

Single Untagged Mode

- If a RADIUS timeout action is not configured, the MAC session is cleared, and a new authentication is initiated.
- If an authentication timeout action is configured as "failure," the behavior is the same as mentioned in Authentication Failure Scenarios on page 194.
- If an authentication timeout action is configured as "success," the client is authenticated in the auth-default VLAN or the previously authenticated VLAN, depending on the following conditions:
  - If a RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the auth-default VLAN.
  - If a RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation. The VLAN can be either a dynamic untagged or tagged VLAN.
- If a RADIUS timeout action is configured as "critical-vlan," the action is implemented based on the following conditions:
  - If it is the first client authenticated on the port, the new client is authenticated in the critical VLAN.
  - If the previous sessions are in the auth-default VLAN or RADIUS-assigned VLAN, the new client is blocked.
  - If the previous sessions are in the restricted VLAN or guest VLAN, the MAC address is blocked.
  - If the previous sessions are in the critical VLAN, the client is authenticated in the critical VLAN.
  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation. The VLAN can be either a dynamic untagged or tagged VLAN.

Multiple Untagged Mode

- If a RADIUS timeout action is not configured, the MAC session is cleared, and a new authentication is initiated.
- If a RADIUS timeout action is configured as "failure," the behavior is the same as mentioned in Authentication Failure Scenarios on page 194.
- If a RADIUS timeout action is configured as "success," the action is implemented based on the following conditions:
  - If a RADIUS timeout occurs during the first authentication attempt, the client is authenticated in the auth-default VLAN.

  - If a RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation. The VLAN can be either a dynamic untagged or tagged VLAN.
  - For MAC authentication, if the authentication is initiated by a tagged packet, the client is authenticated in the VLAN ID carried by the packet tag value.
- If a RADIUS timeout action is configured as "critical-vlan," the action is implemented based on the following conditions:
  - The client is moved to the critical VLAN.
  - If the RADIUS timeout occurs during reauthentication of a previously authenticated client, the client is retained in the previously authenticated VLAN with the existing dynamic ACL allocation. The VLAN can be either a dynamic untagged or tagged VLAN.
  - For MAC authentication, if the authentication is initiated by a tagged packet, the client is blocked in the VLAN ID carried by the packet tag value.

Authentication Client Timeout Scenarios (No Response to EAP Packets)

The dynamic VLAN assignment when the client does not respond to EAP packets is applicable only to 802.1X authentication. VLAN assignment when the client does not respond to the EAP packets depends on whether the authentication mode of the port is single untagged mode (the default) or multiple untagged mode.

Single Untagged Mode

- If there is no response from the client for EAP packets and if the guest VLAN is not configured, the behavior is the same as mentioned in Authentication Failure Scenarios on page 194.
- If the guest VLAN is configured:
  - If it is the first client on the port, the client is authenticated in the guest VLAN.
  - If the previous sessions are in a different RADIUS-assigned VLAN, the client is blocked.
  - If the previous sessions are in the guest VLAN, the new client is permitted in the guest VLAN.
  - If the previous sessions are in a critical VLAN or restricted VLAN, the client is blocked.
  - If the previous sessions are in the guest VLAN and the new client is dot1x-capable, then existing guest MAC sessions are cleared before permitting the new client in the auth-default VLAN or RADIUS-assigned VLAN.

Multiple Untagged Mode

If there is no response from the client for EAP packets and if the guest VLAN is configured, the port is moved to the guest VLAN; otherwise, the failure action is carried out.

Automatic Removal of Dynamic VLAN Assignments for 802.1X and MAC Authenticated Ports

By default, the ICX device removes any association between a port and a dynamically assigned VLAN when authenticated MAC sessions for that tagged or untagged VLAN have expired on the port. Thus, RADIUS-specified VLAN assignments are not saved to the ICX running-config file. When the **show running-config** command is used during a session, dynamically assigned VLANs are not displayed, although they can be displayed with the **show vlan** and **show authentication sessions** commands.

Dynamic ACLs in Authentication

After successful authentication, different network policies can be applied to restrict the way the client accesses network resources. The 802.1X authentication and MAC authentication implementations support dynamically applying IPv4 ACLs and IPv6 ACLs to a port, based on information received from an authentication server.

When a client or supplicant is authenticated, the authentication server (the RADIUS server) sends the authenticator (the RUCKUS ICX device) a RADIUS Access-Accept message that grants the client access to the network. The RADIUS Access-Accept message contains attributes set for the user in the user profile for 802.1X authentication or the device profile for MAC authentication on the RADIUS server.

ACL Filters Downloaded from RADIUS

The Access-Accept message received from the RADIUS authentication server may include Filter-ID 11 as an attribute, in which case the name or number of an ACL already configured on the ICX device can be applied to the Flexible authentication session. The Access-Accept message received from the RADIUS authentication server may also contain the RADIUS-assigned filter attribute Foundry-NAS-Ipv4-Filter-Rule (Filter-id 12) or Foundry-NAD-IPv6-Filter-Rule (Filter-id 13), in which case, the RUCKUS ICX device can use a filter statement included with the attribute to apply an IPv4 or IPv6 ACL to the authenticated port. The RADIUS-assigned ACL filter statement is often referred to as a downloadable ACL, or dACL. The applied dACL can be an extended IPv4 ACL or an IPv6 ACL.

A dACL, like an ACL specified by a Filter-ID 11 attribute, is applied to the port for as long as the client is connected to the network. The applied ACL is removed from the port when the client logs out, the port goes down, or the MAC session ages out.

It is possible for an Access-Accept message received from the RADIUS authentication server to contain multiple RADIUS-assigned attributes (a combination of Filter-id 11, Foundry-NAS-Ipv4-Filter-Rule, or Foundry-NAS-IPv6-Filter-Rule attributes). When a Filter-id 11 attribute containing an ACL name or number and a Foundry-NAS-Ipv4-Filter-Rule or Foundry-NAS-IPv6-Filter-Rule dACL are present in the same RADIUS Access-Accept message, the Filter-id 11 content is disregarded.

The VSAs listed in the following table support downloadable ACL (dACL) filter statements.

TABLE 27 RUCKUS VSAs for RADIUS-Downloadable ACLs

Attribute Name	Attribute ID	Data Type	Description
Foundry-NAS-Ipv4-Filter-Rule	12	string	Includes an IPv4 ACL filter statement that is received and programmed on the ICX device for the duration of the RADIUS session.
Foundry-NAS-Ipv6-Filter-Rule	13	string	Includes an IPv6 ACL filter statement that is received and programmed on the ICX device for the duration of the RADIUS session.

The VSAs should be added into the RADIUS server as new attributes by importing an updated Foundry .dictionary file. The .dictionary file is available at the following URL:

https://github.com/redBorder/freeradius/blob/master/share/dictionary.foundry

The following examples are RADIUS-assigned attributes that contain ACL filter statements. The same IPv4 or IPv6 ACL filter is programmed on the ICX device for the relevant session. The user can send multiple filters by repeating the VSA ID followed by a complete filter-rule statement.

- Foundry-NAS-Ipv4-Filter-Rule = "permit udp 20.20.20.0/24 20.20.20.0/24"
- Foundry-NAS-Ipv6-Filter-Rule = "permit ipv6 host 20:20::22/64 any"

Considerations for Downloadable ACLs Configured on a RADIUS Server

The following restrictions and considerations apply to dynamic IPv4 extended ACLs and IPv6 ACLs downloaded to an ICX device from a RADIUS server:

- A maximum of seven filter statements can be sent.
- ACLS downloaded from RADIUS can be applied only to inbound traffic. Outbound ACLs are not supported.
- IPv4 standard ACLs are not supported for RADIUS download.

- In creating ACL statements for delivery from RADIUS, the user must provide complete ACL filter statements, with no abbreviations.

  Example

  Wrong: per ip an an

  Correct: permit ip any any
- ACLs with mirror or log options are not supported.
- Domain names are not supported in ACL filters. Use matching IP addresses instead.
- Well-known protocol or port names such as PIM or SSH are not supported in RADIUS-downloaded ACLs, and their use may result in an error state. Instead, well-known protocol or port numbers must be used in the filter statements.
- RADUIS-assigned ACL filter statements are given priority if an ACL ID is sent in the same RADIUS Accept message.

Dynamically Applying Existing ACLs in Flexible Authentication

When a port is authenticated, an IPv4 ACL or IPv6 ACL that exists in the running-config file on the RUCKUS ICX device can be dynamically applied to the port. To do this, you configure the Filter-Id (type 11) attribute on the RADIUS server.

The standard RADIUS Filter-id attribute defined in RFC 2865 for IP ACL is shown in the following table.

TABLE 28 Standard RADIUS Attributes for an IP ACL

Attribute Name	Attribute ID	Data Type	Description
Filter-id	11	String	IPv4 or IPv6 ACL ID or name as configured on the RUCKUS device

The syntax in the following table is used to configure the Filter-id attribute to refer to an IPv4 or IPv6 ACL.

TABLE 29 Syntax for Configuring the Filter-Id Attribute

Syntax	Description
ip.*number***.in**, **ip.***name***.in**	Applies the specified numbered or named IPv4 ACL to the authenticated port in the inbound direction.
ip.*number***.out**, **ip.***name***.out**	Applies the specified numbered or named IPv4 ACL to the authenticated port in the outbound direction.
ip6.*number***.in**, **ip6.***name***.in**	Applies the specified numbered or named IPv6 ACL to the authenticated port in the inbound direction.
ip6.*number***.out**, **ip6.***name***.out**	Applies the specified numbered or named IPv6 ACL to the authenticated port in the outbound direction.

The following table lists examples of values that you can assign to the Filter-Id attribute on the RADIUS server to refer to IP ACLs configured on a RUCKUS ICX device.

TABLE 30 Sample Filter-Id Attribute Values on the RADIUS Server

Possible Values for the Filter-Id Attribute on the RADIUS Server	ACL Configured on the RUCKUS Device
ip.102.in	ip access-list extended 102 sequence 10 permit ip any any
ip.fdry_filter.in	ip access-list extended fdry_filter sequence 10 permit icmp any any

Configuration Guidelines for Dynamic ACLs Configured on an ICX Device

The following restrictions apply to dynamic IPv4 ACLs and IPv6 ACLs configured on an ICX device:

- Dynamic ACLs are supported only for inbound traffic.
- For dynamic ACLs, the ND packet hop-limit check is supported only for inbound traffic.
- By default, Flexible authentication supports hop-limit checks for the following types of packets using implicit rules:
  - ND
  - NS
  - RS
- If the RADIUS-provided egress ACL contains an explicit "permit" filter for ND, NS, RS, or RA packets, the hop-limit check is still performed on outbound traffic.
- The name in the Filter-Id attribute is case-sensitive.
- When an ACL is created, make sure to immediately add at least one rule to the ACL.
- IPv4 ACLs can be used in Web authentication. IPv6 ACLs are not supported for Web authentication.
- Both dynamically assigned inbound and outbound IPv4 ACLs and IPv6 ACLs can be applied together. If both IPv4 ACLs and IPv6 ACLs are applied, only one inbound type or one outbound type of each filter is allowed.
- A maximum of one IP ACL per client can be configured for IPv4 inbound, IPv4 outbound, IPv6 inbound, and IPv6 outbound on an interface.
- Static ACLs previously configured on the ICX device but not returned by the RADIUS server are not supported on the 802.1X authentication-enabled or the MAC authentication-enabled port.
- Concurrent operation of dynamically applied IP ACLs and static (pre-configured) IP ACLs is not supported.
- Dynamic IP ACL assignment with 802.1X is not supported in conjunction with any of the following features:
  - Protection against ICMP or TCP Denial of Service (DoS) attacks
  - Policy-based routing
  - Web authentication ACL
- Deletion of an ACL ID is not recommended when ACLs are used by Flexible authentication sessions or Web authentication sessions.
- Dynamic ACLs support either one IPv4 address or up to four IPv6 addresses.
- Changing an ACL ID in the RADIUS user profile is not supported during reauthentication.

Support for IP Source Guard Protection

The ICX proprietary IP Source Guard Protection feature, a form of IP Source Guard, can be used in conjunction with Flexible authentication.

When IP Source Guard Protection is enabled using the **authentication source-guard-protection enable** command in interface configuration mode, IP traffic is blocked until the system learns the IP address. Once the IP address is validated, traffic containing that source IP address is permitted.

> **NOTE**
> In Flexible authentication, IP Source Guard Protection is applicable only for IPv4 traffic.

When a Flexible authentication session is created on a port that has IP Source Guard Protection enabled, the session either applies a dynamically created IP Source Guard ACL entry or uses the dynamic IP ACL assigned by the RADIUS server. If a dynamic IP ACL is not assigned, the session uses the IP Source Guard ACL entry. The IP Source Guard ACL entry can be **permit ip** *secure- ip***any**, where *secure- ip* is obtained from the ARP Inspection table or from the DHCP Secure table. The DHCP Secure table includes DHCP Snooping and Static ARP Inspection entries.

The IP Source Guard ACL entry is not written to the running-config file. However, you can view the configuration using the **show authentication sessions** command.

NOTE
The secure MAC-to-IP mapping is assigned at the time of authentication and remains in effect as long as the session is active. The existing session is not affected if the DHCP Secure table is updated after the session is authenticated and while the session is still active. Change of IP address is supported.

The IP Source Guard ACL permit entry is removed when the session expires or is cleared.

For more information about IP Source Guard, refer to "IP Source Guard" in the "DHCPv4" chapter of the *RUCKUS FastIron DHCP Configuration Guide*.

Configuring Flexible Authentication

Flexible authentication requires some prerequisite tasks that must be performed before executing Flexible authentication configurations at the global and interface levels. Flexible authentication configurations also include 802.1X authentication-specific and MAC authentication-specific configurations.

Flexible Authentication Configuration Prerequisites

Before you configure Flexible authentication, you must establish communication between the devices and the authentication server. The following items cover the configuration steps that are required before you configure Flexible authentication:

- Configure the ICX device interaction with the authentication server by configuring an authentication method list for 802.1X and specifying RADIUS as an authentication method. The method list takes care of 802.1X authentication and MAC authentication. For more information, refer to AAA Operations for RADIUS on page 39.

```
device(config)# aaa authentication dot1x default radius
```

- Configure the RADIUS server to authenticate access to the RUCKUS ICX device. For more information, refer to AAA Operations for RADIUS on page 39.

```
device(config)# radius-server host 10.20.64.208 auth-port 1812 acct-port 1813 default key secretkey
dot1x mac-auth
```

- After successful authentication, the client is moved to the RADIUS-assigned VLAN. Configure a VLAN as the auth-default VLAN to enable authentication. When any port is enabled for 802.1X 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 VLANs based on authentication failure and timeout scenarios.

  NOTE
  RUCKUS recommends that you configure VLANs you plan to use before assigning them as auth-default-vlan or a special VLAN (guest VLAN, restricted VLAN, or critical VLAN).

```
device(config)# vlan 20 name auth-default-vlan
```

- After a successful authentication, user access can be limited by ACLs. ACLs must be preconfigured on the ICX device, and the RADIUS server can return the ACL ID or name. If the ACL matches the ACL configured on the device, it is applied to the port.

```
device(config)# ip access-list extended 100
device(config-ext-ipacl-100)# permit ip any any
```

  NOTE
  The source IP must be either the user's IP address or “any” because the RUCKUS ICX device dynamically learns the IP addresses of the clients (source). The destination network is user-configurable.

For more information on ACL configuration, refer to IPv4 ACLs on page 89. For more information about dynamic ACL assignment, refer to Dynamic ACLs in Authentication on page 196.

- If any of the clients need to be statically authenticated or denied access, the MAC addresses of such clients can be configured using the **authentication filter** command in interface configuration sub-mode.

  ```
  device(config-if-e1000-1/1/1)# authentication filter permit/deny xxxx.xxxx.xxxx FFFF.FFFF.FFFF
  ```

  For additional information, refer to Configuring Flexible Authentication on an Interface on page 203 and the *RUCKUS FastIron Command Reference*.

Configuring Flexible Authentication Globally

The following steps configure Flexible authentication at the global level.

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

   ```
   device# configure terminal
   ```

2. Enter the **authentication** command to enter authentication configuration mode.

   ```
   device(config)# authentication
   ```

   All the global authentication configurations are available in the authentication configuration mode.

3. (Optional) Enter the **auth-order mac-auth dot1x** command to change the sequence of authentication methods to MAC authentication followed by 802.1X authentication if required.

   NOTE
   If the 802.1X authentication and MAC authentication methods are enabled on the same port, by default, the authentication sequence is set to perform 802.1X authentication followed by MAC authentication.

   ```
   device(config-authen)# auth-order mac-auth dot1x
   ```

4. Enter the **auth-default-vlan** command to configure the authentication default VLAN (auth-default VLAN).

   NOTE
   The auth-default VLAN must be configured to enable Flexible authentication before enabling 802.1X authentication or MAC authentication.

   NOTE
   Create the VLAN you plan to use before configuring it as the auth-default VLAN.

   NOTE
   Ports that are members of the auth-default VLAN cannot be enabled for Flexible authentication. Likewise, ports that are enabled for Flexible authentication cannot be added manually to the auth-default VLAN.

   ```
   device(config-authen)# auth-default-vlan 2
   ```

   All ports are moved to the auth-default VLAN as MAC-VLAN members when 802.1X authentication or MAC authentication is enabled. The client remains in the auth-default VLAN if the RADIUS server does not return VLAN information upon authentication or if the RADIUS timeout action is specified as "success" when the RADIUS server is not reachable.

5. (Optional) Enter the **restricted-vlan** command to configure the restricted VLAN.

   ```
   device(config-authen)# restricted-vlan 4
   ```

   When a restricted VLAN is configured, you can configure the authentication failure action as moving the client to the restricted VLAN. If a restricted VLAN is not configured, when authentication fails, the client's MAC address is blocked in the hardware.

6. (Optional) Enter the **auth-fail-action** command to move the port to the restricted VLAN after authentication failure.

```
device(config-authen)# auth-fail-action restricted-vlan
```

7. (Optional) Enter the **critical-vlan** command to configure the VLAN in which the port should be placed when the RADIUS server times out while authenticating or reauthenticating.

```
device(config-authen)# critical-vlan 20
```

8. (Optional) Enter the **auth-timeout-action** command to move the port to the critical VLAN after RADIUS authentication timeout.

```
device(config-authen)# auth-timeout-action critical-vlan
```

9. (Optional) Enter the **auth-mode** command to enable multiple untagged mode, which allows Flexible authentication-enabled ports to be members of multiple untagged VLANs, or single-host and multiple-host mode.

NOTE
By default, a Flexible authentication-enabled port can be a member of only one untagged VLAN (single-untagged mode), and other clients that are authenticated with different dynamic untagged VLANs are blocked.

```
device(config-authen)# auth-mode multiple-untagged
```

10. (Optional) Enter the **disable-aging permitted-mac-only** command to prevent the permitted MAC sessions from being aged out. Or enter the **disable-aging denied-mac-only** to prevent the denied MAC sessions from aging out.

```
device(config-authen)# disable-aging permitted-mac-only
```

NOTE
You can disable aging of either the permitted (authenticated and restricted) sessions or the denied sessions. Once configured, MAC addresses that are authenticated or denied by a RADIUS server are not aged out if no traffic is received from the MAC address for a certain period of time. Aging for a permitted or non-blocked MAC address occurs in two phases, MAC aging and software aging. The MAC aging interval is configured using the **mac-age-time** command. By default, **mac-age-time** is set to 300 seconds. After the normal MAC aging period for permitted clients (or clients in a restricted VLAN), the software aging period begins. The **max-sw-age** command is used to specify the software aging period and by default is set to 120 seconds. After the software aging period ends, the client session ages out and is removed from the session table.
If during software aging, traffic is received from a client, the MAC address of the client is updated in the hardware table, and the client continues to communicate. Software aging is not applicable for blocked MAC addresses.
The hardware aging period for blocked MAC addresses is set to 70 seconds by default, and it can be configured using the **max-hw-age** command. Once the hardware aging period ends, the blocked MAC address ages out and can be authenticated again if the ICX device receives traffic from the MAC address.

11. (Optional) Enter the **max-hw-age** command to configure the hardware aging period for denied MAC addresses.

```
device(config-authen)# max-hw-age 160
```

12. (Optional) Enter the **max-sw-age** command to configure the software aging period.

```
device(config-authen)# max-sw-age 160
```

13. (Optional) Enter the **max-sessions** command to configure the number of clients allowed on a port, the default being 2.

```
device(config-authen)# max-sessions 32
```

14. (Optional) Enter the **re-authentication** command to configure the ICX device to periodically reauthenticate the authenticated clients.

```
device(config-authen)# re-authentication
```

NOTE
When the periodic reauthentication is enabled, the device reauthenticates clients every 3600 seconds (one hour) by default. The reauthentication interval configured using the **reauth-period** command takes precedence.

15. (Optional) Enter the **reauth-period** command to configure the interval at which authenticated clients are reauthenticated. The default period is an hour, or 3600 seconds.

```
device(config-authen)# reauth-period 2000
```

16. (Optional) Enter the **authentication reauth-timeout** command to configure the interval at which non-authenticated (timed out) clients with restricted, critical, or auth-default access are reauthenticated. The default period is 300 seconds.

NOTE
Reauthentication is supported for restricted, critical access, and auth-default VLANs as well as the BLOCKED state VLAN (4092). It is not supported for guest access.

NOTE
A session is moved to the BLOCKED state VLAN if the timeout-action is "failure" and no restricted VLAN has been configured.

```
device(config-authen)# authentication reauth-timeout 120
```

17. (Optional) Specify the IPv4 or IPv6 ingress or egress ACLs to be applied when clients are non-authenticated for various reasons, such as auth-failure, auth-timeout, and access becomes restricted, critical, or guest. Authenticated clients can be assigned ACLs by the RADIUS server.

```
device(config-authen)# default-acl ipv4 guest in
```

18. (Optional) Specify the voice VLAN to be used to add the port as tagged in voice VLAN when it is not provided by the RADIUS server and when the clients are non-authenticated for various reasons, such as auth-failure and auth-timeout.

```
device(config-authen)# voice-vlan 200
```

Configuring Flexible Authentication on an Interface

The following steps configure Flexible authentication at the interface level.

NOTE
Configuration at the interface level overrides related configuration at the global level. The global configuration is still applicable to other ports that do not have a per-port configuration. Refer to Configuring Flexible Authentication Globally on page 201 for more information.

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command to enter interface configuration mode.

```
device(config)# interface ethernet 1/1/1
```

3. (Optional) Enter the **authentication auth-order mac-auth dot1x** command to change the sequence of authentication to MAC authentication followed by 802.1X authentication if required.

```
device(config-if-e1000-1/1/1)# authentication auth-order mac-auth dot1x
```

4. Enter the **authentication auth-default-vlan** command to configure the authentication default VLAN (auth-default VLAN).

   NOTE
   Ports that are members of the auth-default VLAN cannot be enabled for Flexible authentication. Likewise, ports that are enabled for Flexible authentication cannot be added manually to the auth-default VLAN.

   NOTE
   Create the VLAN you plan to use before configuring it as the auth-default VLAN.

```
device(config-if-e1000-1/1/1)# authentication auth-default-vlan 30
```

5. (Optional) Enter the **authentication auth-mode** command to enable the multiple untagged mode on a specific Flexible authentication-enabled port and allow it to be a member of multiple untagged VLANs.

```
device(config-if-e1000-1/1/1)# authentication auth-mode multiple-untagged
```

6. (Optional) Enter the **authentication disable-aging permitted-mac-only** or the **authentication disable-aging denied-mac-only** command to prevent the permitted or denied MAC sessions from being aged out from a port.

```
device(config-if-e1000-1/1/1)# authentication disable-aging permitted-mac-only
```

7. (Optional) Enter the **authentication max-sessions** command to specify the maximum limit of authenticated MAC sessions on an interface.

```
device(config-if-e1000-1/1/1)# authentication max-sessions 32
```

8. (Optional) Enter the **authentication dos-protection** command to enable Denial of Service (DoS) authentication protection on an interface.

```
device(config-if-e1000-1/1/1)# authentication dos-protection mac-limit 256
```

   NOTE
   You can also configure the RUCKUS ICX device to limit the rate of authentication attempts sent to the RADIUS server.

9. (Optional) Enter the **authentication source-guard-protection** command to enable IP Source Guard Protection along with authentication on an interface.

```
device(config-if-e1000-1/1/1)# authentication source-guard-protection enable
```

10. (Optional) Enter the **authentication voice-vlan** command to specify the voice VLAN to be used to add the port as tagged in the voice VLAN when it is not provided by the RADIUS server and when the clients are non-authenticated for various reasons, such as auth-failure and auth-timeout.

```
device(config-if-e1000-1/1/1)# authentication voice-vlan 300
```

11. (Optional) Enter the **authentication allow-tagged** command to allow tagged packet processing when the port is not tagged, which may be the case when multiple VMs are connected to the port so that they can be authenticated with MAC authentication, and automatic tagging of the port helps. This option is disabled by default.

```
device(config-if-e1000-1/1/1)# authentication allow-tagged
```

12. (Optional) Enter the **authentication filter** command to set up 802.1X authentication override. On the same line, enter a **permit** statement with the appropriate MAC address and mask so that the MAC addresses specified do not go through authentication.

    ```
    device(config-if-e1000-1/1/1)# authentication filter permit 0001.1234.1234 ffff.ffff.ffff 10
    ```

    The source MAC addresses defined using the **authentication filter** command are considered pre-authenticated and are not subject to authentication. A client can be authenticated in an untagged VLAN or tagged VLAN using the authentication filter. If the authentication filter has a tagged VLAN configuration, the clients are authenticated in the auth-default VLAN and the tagged VLAN provided in the authentication filter statement (VLAN ID 10 in the previous example). The clients authorized in the auth-default VLAN allow both untagged and tagged traffic.

Enabling 802.1X Authentication

The following steps are for enabling and activating 802.1X authentication and for configuring certain 802.1X-specific commands.

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

   ```
   device# configure terminal
   ```

2. Enter the **authentication** command to enter authentication mode.

   ```
   device(config)# authentication
   ```

3. Enter the **dot1x enable** command to enable 802.1X authentication.

   NOTE
   Ports that are members of the auth-default VLAN cannot be enabled for Flexible authentication. Likewise, ports that are enabled for Flexible authentication cannot be added manually to the auth-default VLAN.

   ```
   device(config-authen)# dot1x enable
   ```

4. Enter the **dot1x enable**{**all**|**ethernet***unit/slot/port*[**to***unit/slot/port*]} command to enable 802.1X authentication on all interfaces, a single interface, or a specific set of interfaces.

   ```
   device(config-authen)# dot1x enable all
   ```

5. Enter the **dot1x port-control auto** command to set the controlled port in the unauthorized state until authentication takes place between the client and the authentication server.

   The following example configures the command globally.

   ```
   device(config-authen)# dot1x port-control auto all
   ```

   The following example configures the command on a single interface. (Interface configuration overrides global configuration if they differ.)

   ```
   device(config-authen)# dot1x port-control auto ethernet 1/1/1
   ```

   Once the client passes authentication, the port becomes authorized. This activates authentication on an 802.1X-enabled interface. The controlled port remains in the authorized state until the client logs off.

6. (Optional) Enter the **dot1x guest-vlan** command to configure the VLAN into which the port should be placed when the client's response to the dot1x requests for authentication times out.

   ```
   device(config-authen)# dot1x guest-vlan
   ```

7. (Optional) Configure the timeout parameters that determine the time interval for client reauthentication and EAP retransmissions using the following commands:
   - Enter the **dot1x timeout quiet-period** command to configure the amount of time the ICX device should wait before reauthenticating the client.

     ```
     device(config-authen)# dot1x timeout quiet-period 30
     ```

   - Enter the **dot1x timeout tx-period** command to configure the amount of time the ICX device should wait before retransmitting EAP-Request/Identity frames to the client.

     ```
     device(config-authen)# dot1x timeout tx-period 30
     ```

   - Enter the **dot1x timeout supplicant** command to configure the amount of time the ICX device should wait before retransmitting RADIUS EAP-Request/Challenge frames to the client.

     ```
     device(config-authen)# dot1x timeout supplicant 30
     ```

   Based on the timeout parameters, the client is reauthenticated, and EAP-Request/Identity frames and EAP-Request/Challenge frames are retransmitted.
8. (Optional) Enter the **dot1x max-reauth-req** command to configure the maximum number of times EAP-Request/Identity frames are sent for reauthentication after the first authentication attempt.

   ```
   device(config-authen)# dot1x max-reauth-req 4
   ```

   If no EAP Response/Identity frame is received from the client after the specified number of EAP-Request/Identity frame retransmissions, the device restarts the authentication process with the client.
9. (Optional) Enter the **dot1x max-req** command to configure the maximum number of times EAP-Request/Challenge frames are retransmitted when an EAP Response/Identity frame is not received from the client.

   ```
   device(config-authen)# dot1x max-req 3
   ```

10. (Optional) Enter the **dot1x macauth-override** command to configure the device to perform MAC authentication after 802.1X authentication, if 802.1X authentication fails for the clients.

    NOTE
    This command is applicable only when the authentication sequence is configured as 802.1X authentication followed by MAC authentication.

    ```
    device(config-authen)# dot1x macauth-override
    ```

Enabling MAC Authentication

The following steps enable MAC authentication and include certain Flexible authentication configurations specific to MAC authentication.

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

   ```
   device# configure terminal
   ```

2. Enter the **authentication** command to enter authentication mode.

   ```
   device(config)# authentication
   ```

3. Enter the **mac-authentication enable** command to enable MAC authentication.

   NOTE
   Ports that are members of the auth-default VLAN cannot be enabled for Flexible authentication. Likewise, ports that are enabled for Flexible authentication cannot be added manually to the auth-default VLAN.

   ```
   device(config-authen)# mac-authentication enable
   ```

4. Enter the **mac-authentication enable** { **all** | **ethernet** *stack/slot/port* } command to enable MAC authentication on all interfaces or a specific interface.

   ```
   device(config-authen)# mac-authentication enable all
   ```

5. (Optional) Enter the **mac-authentication password-format** command to configure the format in which the MAC address is sent to the RADIUS server for authentication.

   By default, the MAC address is sent to the RADIUS server in lowercase in the xxxxxxxxxxxx format. As an option, you can change the address to uppercase. You can specify one of the following formats:

   - xx-xx-xx-xx-xx-xx
   - xx:xx:xx:xx:xx:xx
   - xxxx.xxxx.xxxx
   - xxxxxxxxxxxx

   ```
   device(config-authen)# mac-authentication password-format xx-xx-xx-xx-xx-xx upper-case
   ```

6. (Optional) Enter the **mac-authentication password-override** command to specify a user-defined password instead of the MAC address for MAC authentication.

   NOTE
   The password can contain up to 32 alphanumeric characters but must not include blank spaces.

   ```
   device(config-authen)# mac-authentication password-override ts54fs
   ```

7. (Optional) Enter the **mac-authentication dot1x-disable** command to configure the device not to perform 802.1X authentication after MAC authentication when MAC authentication succeeds for the client. This is enabled by default, unless overruled by the RADIUS server through a dot1x-enable attribute.

   NOTE
   This command is applicable only when the authentication sequence is configured as MAC authentication followed by 802.1X authentication.

   ```
   device(config-authen)# mac-authentication dot1x-disable
   ```

8. (Optional) Enter the **mac-authentication dot1x-override** command to configure the device to perform 802.1X authentication after MAC authentication, if MAC authentication fails for the client.

   NOTE
   This command is applicable only when the authentication sequence is configured as MAC authentication followed by 802.1X authentication.

   ```
   device(config-authen)# mac-authentication dot1x-override
   ```

Excluding the RADIUS Server for Login Features

You can specify whether the RADIUS server can be used for login features such as Telnet, SSH, console, EXEC, or Web-management AAA. The following task excludes the RADIUS server for all login features.

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

   ```
   device# configure terminal
   ```

2. Use the **radius-server host** command, specifying an IP address, and using the **auth-port** *port-num*, **acct-port** *port-num*, **default**, **key** *key-string*, and **no-login** parameters.

   ```
   device(config)# radius-server host 10.26.67.13 auth-port 1812 acct-port 1813 default key ruckus no-
   login
   ```

   The example configures default UDP ports for authorization and accounting. The RADIUS key for the server is configured as “ruckus." The **no-login** keyword is configured so that this RADIUS server cannot be used for Telnet, SSH, console, EXEC, or Web-management AAA.

The following example uses the RADIUS server for Flexible authentication modules.

```
device# configure terminal
device(config)# radius-server host 10.26.67.13 auth-port 1812 acct-port 1813 default key ruckus mac-auth
dot1x
```

The following example uses the RADIUS server for Flexible authentication and excludes the use of the server for login features.

```
device# configure terminal
device(config)# radius-server host 10.26.67.13 auth-port 1812 acct-port 1813 default key ruckus mac-auth
dot1x no-login
```

Displaying Authentication Information

You can display the following authentication information:

- The authentication configuration on the device and on individual ports
- Statistics about the EAPOL frames passing through the device
- Hosts or user session information
- Authentication-enabled ports dynamically assigned to a VLAN
- User-defined and dynamically applied IP ACLs currently active on the device

Displaying Configuration

The output of the **show authentication configuration** command indicates the global configuration details for authentication, as shown in the following example.

```
device# show authentication configuration
Auth:
Auth Order                          : mac-auth dot1x
Default VLAN                        : 10
Default Voice VLAN                  : Not configured
Auth Mode                           : Multiple Untagged Mode
Restricted VLAN                     : Not configured
Critical VLAN                       : Not configured
Auth Failure Action                 : Block traffic
Auth Timeout Action                 : Treat as a successful authentication
MAC Session Aging                   : Enabled
Re-authentication                   : Enabled
Reauth-period                       : 180 seconds
```

```
Reauth-timeout                    : 120 seconds
Session Max SW-Age                : 120 seconds
Session Aax HW-Age                : 70 seconds
Max Sessions                      : 2
MAC-Auth:
Status                            : Enabled
802.1X Override                   : Enabled
Password Override                 : Disabled
Password Format                   : xxxx.xxxx.xxxx
802.1X:
Status                            : Enabled
Protocol Version                  : 1
PAE Capability                    : Authenticator Only
MAC-Auth Override                 : Disabled
Guest VLAN                        : Not configured
Quiet-period                      : 60 seconds
TX-period                         : 30 seconds
Supplicant-timeout                : 30 seconds
Max Reauth Requests               : 2
Max Frame Retries                 : 2
```

To display the configuration details for authentication on an interface, enter the **show authentication configuration ethernet** command followed by the port number, as shown in the following example.

```
device# show authentication configuration ethernet 1/1/1
Port 1/1/1 Configuration:
Auth Order                        : mac-auth dot1x
Auth Mode                         : Multiple Untagged Mode
Auth Failure Action               : Block traffic
Auth Timeout Action               : Treat as a successful authentication
DoS Protection                    : Disabled (limit = 512)
Source-guard Protection           : Disabled
Aging                             : Enabled
Max Sessions                      : 32
Reauth-timeout                    : 120 seconds
802.1X Port-Control               : Auto
```

Displaying Statistics

To display statistics for an individual port, enter the **show authentication statistics** command, as shown in the following example.

```
device# show authentication statistics ethernet 1/1/1
Port 1/1/1 Statistics:
802.1X:
RX EAPOL Start:                    0
RX EAPOL Logoff:                   0
RX EAPOL Invalid:                  0
RX EAPOL Total:                    0
RX EAP Resp/Id:                    0
RX EAP Resp other than Resp/Id:    0
RX EAP Length Error:               0
Last EAPOL Version:                0
Last EAPOL Source:                 0000.0050.0B83
TX EAPOL Total:                    217
TX EAP Req/Id:                     163
TX EAP Req other than Req/Id:      0
MAC-Auth:
Accepted Sessions:                 0
Rejected Sessions:                 0
Inprogress Sessions:               0
Attempted Sessions:                0
Number of Errors:                  0
```

The following table describes the information displayed for an interface by the **show authentication statistics** command.

TABLE 31 Output from the show authentication statistics Command

Field	Statistics
RX EAPOL Start	The number of EAPOL-Start frames received on the port
RX EAPOL Logoff	The number of EAPOL-Logoff frames received on the port
RX EAPOL Invalid	The number of invalid EAPOL frames received on the port
RX EAPOL Total	The total number of EAPOL frames received on the port
RX EAP Resp/Id	The number of EAP-Response/Identity frames received on the port
RX EAP Resp other than Resp/Id	The total number of EAPOL-Response frames received on the port that were not EAP-Response/Identity frames
RX EAP Length Error	The number of EAPOL frames received on the port that have an invalid packet body length
Last EAPOL Version	The version number of the last EAPOL frame received on the port
Last EAPOL Source	The source MAC address in the last EAPOL frame received on the port
TX EAPOL Total	The total number of EAPOL frames transmitted on the port
TX EAP Req/Id	The number of EAP-Request/Identity frames transmitted on the port
TX EAP Req other than Req/Id	The number of EAP-Request frames transmitted on the port that were not EAP-Request/Identity frames
Accepted Sessions	The number of MAC authentication sessions accepted
Rejected Sessions	The number of MAC authentication sessions rejected
In Progress Sessions	The number of MAC authentication sessions which are in progress
Attempted Sessions	The number of MAC authentication sessions attempted
Number of Errors	The number of errors encountered while processing sessions

Displaying the Authentication Sessions

Use the **show authentication sessions** command to view details of the 802.1X authentication or MAC authentication sessions, such as the ports, MAC addresses, IP addresses, and VLANs, as shown in the following example.

```
device# show authentication sessions all
-------------------------------------------------------


Port   MAC            IP(v4/v6)            User    VLAN  Auth    Auth    ACL    Session   Age  PAE
       Addr           Addr                 Name          Method  State          Time           State
-------------------------------------------------------------------------------------------------------
-
2/1/25 00aa.aaaa.0000 198.1.1.2            MVDI_1 130   MAUTH   permit  Yes    210       Ena  N/A
2/1/25 00aa.aaaa.0001 fe80::2aa:aaff:feaa  DVDI_1 130   8021.X  permit  Yes    210       Ena  AUTHENTICATED
                      3000::2
                      3000::2
1/1/15 00bb.bbbb.0001 N/A                  DVDI_2 230   8021.X  permit  None   500       Ena  AUTHENTICATED
1/1/10 0010.9400.1101 N/A                  MVDI_2 330   MAUTH   permit  None   410       Ena  N/A
```

The following example displays sessions for a specific interface.

```
device(config)# show authentication sessions ethernet 2/1/25
-------------------------------------------------------------------
Port   MAC            IP(v4/v6)            User    VLAN  Auth    Auth    ACL    Session   Age  PAE
       Addr           Addr                 Name          Method  State          Time           State
-------------------------------------------------------------------------------------------------------
-
2/1/25 00aa.aaaa.0000 198.1.1.2            MVDI_1  130   MAUTH   permit  Yes    210       Ena  N/A
2/1/25 00aa.aaaa.0001 fe80::2aa:aaff:feaa  DVDI_1  130   8021.X  permit  Yes    210       Ena  AUTHENTICATED
```

The following example displays sessions for a specific stack unit.

```
device(config)# show authentication sessions unit 1
-----------------------------------------------------------------
Port   MAC            IP(v4/v6)           User    VLAN  Auth    Auth    ACL    Session   Age  PAE
       Addr           Addr                Name          Method  State          Time           State
-------------------------------------------------------------------------------------------------------
-
1/1/15 00bb.bbbb.0001 N/A                 DVDI_2  230   8021.X  permit  None   500       Ena  AUTHENTICATED
1/1/10 0010.9400.1101 N/A                 MVDI_2  330   MAUTH   permit  None   410       Ena  N/A
```

The following example displays a brief description of authentication sessions. Notice that Port 1/1/7 shows 4 authentication attempts, 1 authorized user, and 3 denied users, with the Untagged VLAN Type of RADIUS-VLAN. This is the case when at least one of the users on the same single untagged VLAN is using a RADIUS-returned VLAN. (The critical VLAN, restricted VLAN, guest VLAN, and dynamically allocated VLAN can be the same, and any of these can be assigned by the RADIUS server.)

```
device# show authentication sessions brief
-------------------------------------------------------------------------------------------------------
Port   Number of         Number of          Number of    Untagged            Dynamic
       Attempted Users   Authorized Users   Denied Users VLAN Type           Port ACL MAC DOT1X
       MAC    DOT1X      MAC    DOT1X
-------------------------------------------------------------------------------------------------------
1/1/7   4      0          1      0           3      0    RADIUS-VLAN         No
1/1/8   1      0          1      0           0      0    Auth-Default-VLAN   No
1/1/9   0      0          0      0           0      0    Multiple            No
1/1/10  0      0          0      0           0      0    Auth-Default-VLAN   No
```

The following example displays a detailed description of an authentication session.

```
device# show authentication sessions detail ethernet 17/1/1
Auth Session Info (Port 17/1/1, MAC a036.9f6e.1fd2) :
State               : Permitted
Auth Method         : 802.1X                Auth Mode           : Single Untagged
VLAN Type           : Radius-VLAN           VLAN                : 200
Voice VLAN          : 0                     PVID                : 0
Tagged VLANs        :
User Name           : joe.user@arris.com
Session Time        : 1381                  Reauth Time         : 2220
Idle Timeout        : 120                   Session Timeout     : 0
Acct session ID     : 2                     PCE Index           : 65535
PAE State           : AUTHENTICATED         Age                 : Disabled
Qos Priority        : 0                     Failure Reason      :
Auth Filter Applied : No                    Tagged              : No
VLAN Add Req State  : Complete              VLAN Del Req State  : Init
Filter Add Req State: Complete              Filter Del Req State : Init
Stale               : No                    Delete Pending      : No
802.1X Enabled      : No                    Session Control     : Self
V4 ACL Applied      : No                    V6 ACL Applied      : No
V4 IN ACL (Session) : acl1                  V4 OUT ACL (Session) : -
V6 IN ACL (Session) : -                     V6 OUT ACL (Session) : -
Client Voice Phone  : No                    Client Wireless AP  : No
802.1X Capable      : Yes
IP Addresses        : 10.176.167.145
V4-IN ACL (Dynamic) : 3928                  V4-OUT ACL (Dynamic) : 0
V6-IN ACL (Dynamic) : 0                     V6-OUT ACL (Dynamic) : 0
V4-IN ACL RefCnt    : 1                     V4-OUT ACL RefCnt    : 0
V6-IN ACL RefCnt    : 0                     V6-OUT ACL RefCnt    : 0
V4 ACL Trap Rule    : Yes                   V6 ACL Trap Rule     : No
Addr Change Count   : 0                     MBV Usage Count      : 1
Radius VLAN RefCnt  : 1

Auth Order          : dot1x, mac-auth       Auth Fail Action     : Restricted VLAN (3)
Auth Timeout Action : Failure               Aging                : Enabled
SG Protection       : Disabled              DOS Protection       : Disabled (limit = 512)
Reauthentication    : Enabled               Reauth Period        : 3600
Reauth Timeout      : 300                   Max Ssessions        : 2
Port Control        : Auto                  Quiet Period         : 60
Supplicant Time     : 30                    Tx Period            : 3
Max Reauth Requests : 2                     Max Frame Retries    : 2
```

Displaying Information About User ACLs

You can display information about the currently active user-defined and dynamically applied ACLs on all the interfaces in the stack, as shown in the following example.

```
device# show authentication acls all
-------------------------------------------------------------------------------
Port     MAC Address       V4 Ingress     V4 Egress      V6 Ingress     V6 Egress
-------------------------------------------------------------------------------
1/1/7    0180.c200.0003    -              -              -              -
1/1/8    0100.c200.0003    10             11             v6in           v6out
1/1/9    0200.c200.0003    100            101            v6in           v6out
```

You can display information about the currently active user-defined and dynamically applied ACLs on a stack unit, as shown in the following example.

```
device# show authentication acls unit 1
-------------------------------------------------------------------------------
Port     MAC Address       V4 Ingress     V4 Egress      V6 Ingress     V6 Egress
-------------------------------------------------------------------------------
1/1/7    0180.c200.0003    -              -              -              -
1/1/8    0100.c200.0003    10             11             v6in           v6out
1/1/9    0200.c200.0003    100            101            v6in           v6out
```

You can display information about the currently active user-defined and dynamically applied ACLs for a specified interface, as shown in the following example.

```
device# show authentication acls ethernet 1/1/9
-------------------------------------------------------------------------------
Port      MAC Address       V4 Ingress     V4 Egress      V6 Ingress     V6 Egress
-------------------------------------------------------------------------------
1/1/9     0200.c200.0003    100            101            v6in           v6out
```

Displaying Dynamically Assigned VLAN Information

The **show vlan ethernet** command displays dynamically assigned VLAN information, as shown in the following example.

```
device# show vlan ethernet 1/1/7
Total PORT-VLAN entries: 8
Maximum PORT-VLAN entries: 1024
Legend: [Stk=Stack-Id, S=Slot]
PORT-VLAN 10, Name Auth-Default, Priority level0, Spanning tree Off
Untagged Ports: None
Tagged Ports: (U1/M1) 48
Mac-Vlan Ports: (U1/M1) 7 8 9
Monitoring: Disabled
device#sh vlan brief ethernet 1/1/7
Port 1/1/7 is a member of 1 VLANs
VLANs 10
MAC VLANs : 10
Tagged VLANs :
```

Clearing Authentication Details

You can clear 802.1X authentication and MAC authentication sessions or statistics counters on individual interfaces or on a range of interfaces, as shown in the following examples.

To clear the statistics counters on an interface, enter a command such as the following.

```
device# clear authentication statistics ethernet 1/1/1
```

To clear the statistics counters on a range of interfaces, enter a command such as the following.

```
device# clear authentication statistics ethernet 1/1/1 to 1/1/10
```

To clear the statistics counters on all the interfaces of a stack unit, enter a command such as the following.

```
device# clear authentication statistics unit 3
```

To clear the statistics counters on all the interfaces in the stack, enter a command such as the following.

```
device# clear authentication statistics
```

To clear the sessions for a specific MAC address, enter a command such as the following.

```
device# clear authentication sessions 0000.0034.abd4
```

To clear the sessions on an interface, enter a command such as the following.

```
device# clear authentication sessions ethernet 1/1/1
```

To clear the sessions on a range of interfaces, enter a command such as the following.

```
device# clear authentication sessions ethernet 1/1/1 to 1/1/8
```

To clear the sessions on all the interfaces of stack unit, enter a command such as the following.

```
device# clear authentication sessions unit 1
```

To clear the sessions on all the interfaces of stack, enter a command such as the following.

```
device# clear authentication sessions
```

HTTP and HTTPS

- Web Authentication Overview 215
- Captive Portal Authentication (External Web Authentication) 216
- Web Authentication Configuration Considerations 220
- Configuring Web Authentication 221
- Prerequisites for Captive Portal Support with RUCKUS Cloudpath 222
- Prerequisites for Configuring Captive Portal with Aruba ClearPass 224
- Prerequisites for Configuring External Web Authentication with Cisco ISE 226
- Prerequisite Configurations on an ICX Switch for Captive Portal Authentication 227
- Creating the Captive Portal Profile for External Web Authentication 227
- Configuring Captive Portal (External Web Authentication) 228
- Web Authentication Mode Configuration 230
- Web Authentication Options 235
- Displaying Web Authentication Information 247
- Image Download over HTTPS 250
- Configuration Download over HTTPS 251
- Configuration Upload over HTTPS 251

Web Authentication Overview

Authentication is important in enterprise networks because the network is considered a secure area: it contains sensitive data and a finite amount of resources. Unauthorized users must be prevented from accessing the network to protect the sensitive data and prevent the unnecessary consumption of resources.

The ideal authentication method blocks unauthorized users at the earliest possible opportunity. For internal enterprise networks, this can be controlled at the edge switch port. Two popular forms of port-based security authentication used at the edge switch are MAC authentication and 802.1x authentication. MAC authentication authenticates the MAC addresses of hosts or users that are attempting to access the network. This type of authentication requires no intervention from the host or user who is attempting to be authenticated. It is easy to use, but it can only authorize hosts; it cannot be used to authorize users. 802.1x authentication can authorize users or hosts. It is more flexible than the MAC authentication method; however, it requires more support, configuration, maintenance and user intervention than MAC authentication.

The RUCKUS Web authentication method provides an ideal port-based authentication alternative to MAC authentication without the complexities and cost of 802.1x authentication. Users gain access to the network by opening a Web browser and entering a valid URL address using HTTP or HTTPS services. Instead of being routed to the URL, the user's browser is directed to an authentication Web page on the ICX switch, or an external authentication server (such as RUCKUS Cloudpath, Aruba ClearPass, or Cisco ISE). The Web page prompts the user to enter a user ID and password or a passcode. The credentials a user enters are used by a trusted source to authenticate the user.

If the authentication is unsuccessful, the appropriate page is displayed on the host browser. The host is asked to try again or call for assistance, depending on what message is configured on the Web page. If the host MAC address is authenticated by the trusted source, a Web page is displayed with a hyperlink to the URL the host originally entered. If the user clicks on the link, a new window is opened and the the user is directed to the requested URL.

While a MAC address is in the authenticated state, the host can forward data through the ICX switch. The MAC address remains authenticated until one of the following events occurs:

- The host MAC address is removed from a list of MAC addresses that are automatically authenticated. (Refer to Permanently Authenticated Hosts on page 236).
- The re-authentication timer expires and the host is required to re-authenticate (Refer to Re-authentication Period on page 237).

- The host has remained inactive for a period of time and the inactive period timer has expired. (Refer to Forcing Re-authentication After an Inactive Period on page 238).
- All the ports on the VLAN on which Web Authentication has been configured are in a down state. All MAC addresses that are currently authenticated are de-authenticated (Refer to Forcing Re-authentication When Ports Are Down on page 238).
- The authenticated client is cleared from the Web Authentication table. (Refer to Clearing Authenticated Hosts from the Web Authentication Table on page 237).

The ICX switch can be configured to automatically authenticate a host MAC address. The host will not be required to login or re-authenticate (depending on the re-authentication period) once the MAC address passes authentication.

A host that is logged in and authenticated remains logged in indefinitely, unless a re-authentication period is configured. When the re-authentication period ends, the host is logged out. A host can log out at any time by pressing the Logout button in the Web Authentication Success page.

NOTE
The host can log out as long as the Logout window (Success page) is visible. If the window is accidentally closed, the host cannot log out unless the re-authentication period ends or the host is manually cleared from the Web Authentication table.

Captive Portal Authentication (External Web Authentication)

Captive portal authentication provides a means to authenticate clients through an external web server. A client that seeks web access to a network is redirected to the authentication web login page hosted on an external network access control (NAC) server (such as RUCKUS Cloudpath, Aruba ClearPass, or Cisco ISE) that is integrated with the RADIUS server.

NOTE
Because the authentication server and web login page reside in an external server, captive portal authentication may also be referred to as external web authentication. These two terms are used interchangeably.

To equip the ICX switch to handle the HTTP redirection mechanism, configuration details specific to the NAC server such as virtual IP address, HTTP or HTTPS protocol port number, and login page details hosted on the NAC server must be specified on the switch. Upon receiving the redirected web access request, the NAC server honors the login page to the client which in turn submits the user login credentials. The NAC server reverts the credentials and sends the username, password, and default URL of the web page to the network-attached storage (NAS) or switch.

NOTE
For details on configuring external captive portal on a NAC server, refer to the user manual for the NAC server being used. ICX switches support RUCKUS Cloudpath, Aruba ClearPass, and Cisco ISE servers.

- For the RUCKUS Cloudpath server, refer to the *Cloudpath ES 5.2 Deployment Guide* (at this URL: https://support.ruckuswireless.com/documents/2006).
- For the Aruba ClearPass server, refer to the Aruba ClearPass Guest User Guide. Refer to the ClearPass Guest 6.4 User Guide, as the version used for validation is 6.4.
- For the Cisco ISE server, refer to Cisco Identity Services Engine documentation.

The ICX switch makes use of the credentials for initiating the authentication process through the RADIUS server, which is integrated with NAC server.

NOTE
The RADIUS server on the ICX switch and the one integrated with the NAC server must have the same configuration.

The RADIUS server validates the user credential information and, if the client is authenticated, the client is redirected to the URL provided by the server. For information about re-authentication and login failure behavior, refer to Re-authentication Period on page 237 and Web Authentication Cycle on page 237.

Captive Portal Profile for External Web Authentication

The Captive Portal profile serves as a template that includes configuration details specific to the external server such as virtual IP address, HTTP or HTTPS protocol port number, and details of the login page hosted on the NAC server.

NOTE
The terms Captive Portal and external Web Authentication are used interchangeably.

The details configured in the Captive Portal profile enable the switch to handle the HTTP redirection mechanism and redirect the client to the login page hosted on the NAC server. The Captive Portal profile is then applied on an external Web Authentication-enabled VLAN.

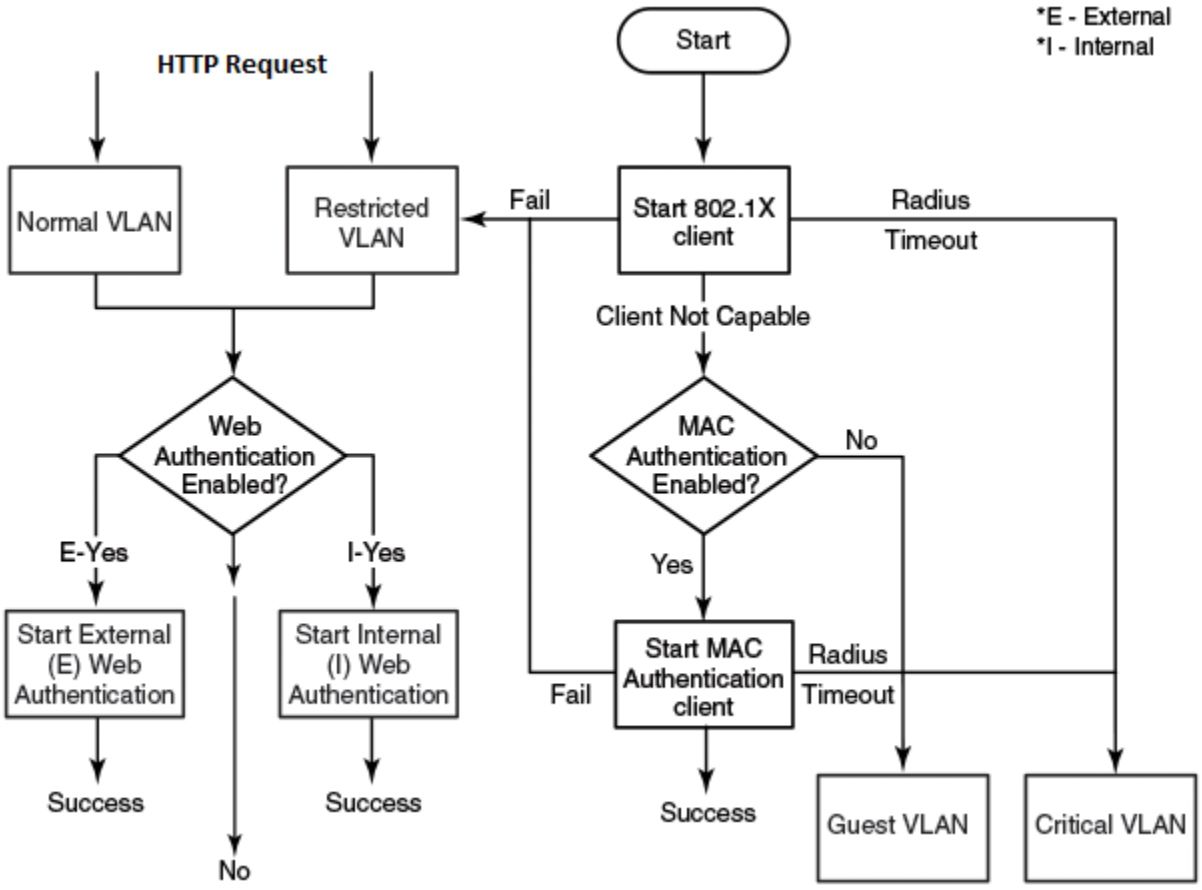
Captive Portal on a VLAN

A Captive Portal can be configured as a fallback authentication method for Flexible authentication (a combination of 802.1X authentication and MAC authentication).

A Captive Portal (sometimes referred to as External Web Authentication) can be enabled on any VLAN associated with Flexible authentication (restricted VLAN, guest VLAN, or critical VLAN). You can also enable a Captive Portal on any VLAN which is independent of Flexible authentication configuration. In either case, the client must send HTTP request for the switch to initiate external Web Authentication.

Figure 13 illustrates the Captive Portal flow on a restricted VLAN configured as part of Flexible authentication and a normal VLAN (any VLAN) which is independent of Flexible authentication configuration.

FIGURE 13 Captive Portal Flow



Dynamic IP ACLs in Web Authentication

After successful authentication, different network policies can be applied to restrict the way the network resources are accessed by the client. Web Authentication implementation (internal and external) support dynamically applying an IP ACL to a port, based on information received from the authentication server.

When a client/supplicant is authenticated, the authentication server (the RADIUS server) sends the authenticator (the RUCKUS device) a RADIUS Access-Accept message that grants the client access to the network. The RADIUS Access-Accept message contains attributes set for the user in the user profile on the RADIUS server.

If the Access-Accept message contains the Filter-Id (type 11) attribute, the RUCKUS device can use information in the attribute to apply an IP ACL filter to the authenticated port. The IP ACL filter applies to the port for as long as the client is connected to the network. The IP ACL filter is removed from the corresponding port when the client logs out.

NOTE
IPv6 ACL is not supported for Web Authentication.

The RUCKUS device uses information in the Filter Id attributes as follows:

- The Filter-Id attribute can specify the number of an existing IP ACL filter configured on the RUCKUS device. In this case, the IP ACL filter with the specified number is applied to the port.

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.
- Dynamically assigned IP ACLs are subject to the same configuration restrictions as non-dynamically assigned IP ACLs.
- Filters are supported for inbound traffic only. Outbound filters are not supported.
- A maximum of one IP ACL per client can be configured in the inbound direction on an interface.
- Static ACLs are not supported with a Web Authentication-enabled port.
- Concurrent operation of a dynamic IP ACL and a static IP ACL is not supported.
- Dynamic IP ACL assignment with Web Authentication is not supported in conjunction with any of the following features:
  - IP Source Guard
  - Rate limiting
  - Protection against ICMP or TCP Denial of Service (DoS) attacks
  - Policy-based routing
  - DHCP snooping
  - ARP inspection
  - Flexible authentication dynamic IPv4 ACLs and MAC ACLs
  - Static MAC ACLs
  - Static IPv4 access list
  - ACL logging

Dynamically Applying Existing ACLs (HTTP and HTTPS)

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 32 Standard RADIUS Attribute for the IP ACL

Attribute Name	Attribute ID	Data Type	Description
Filter-Id	11	String	IPv4 ACL ID or name as configured on the RUCKUS device.

The following table shows the syntax for specifying the IP ACLs on the RADIUS server.

TABLE 33 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 34 Standard RADIUS Attribute for Session Timeout

Attribute Name	Attribute ID	Data Type	Description
Session-Timeout	27	Integer	Session timeout after which session is cleared.

Web Authentication Configuration Considerations

Web Authentication is modeled after other RADIUS-based authentication methods currently available on RUCKUS edge switches. However, Web Authentication requires a Layer 3 protocol (TCP/IP) between the host and the authenticator. Therefore, to implement Web Authentication, you must consider the following configuration and topology configuration requirements:

- Web authentication works only when both the HTTP and HTTPS servers are enabled on the device.
- Web Authentication works only on the default HTTP or HTTPS port.
- The host must have an IP address prior to Web Authentication. This IP address can be configured statically on the host; however, DHCP addressing is also supported.
- If you are using DHCP addressing, a DHCP server must be in the same broadcast domain as the host. This DHCP server does not have to be physically connected to the switch. Also, DHCP assist from a router may be used.
- Web Authentication is not supported on a reserved VLAN.
- Web-authentication does not support 'tagged port.' Tagged port in web-authentication VLAN works only when used as 'trust-port.' Trusted ports (configured using the **trust-port** command) can be tagged to multiple VLANs including web-authentication VLAN as the uplink.
- If a port that is a member of a web-authentication VLAN is also a tagged member of another VLAN, incoming traffic through that port is blocked, including traffic intended for any VLAN to which it belongs. In a network environment that requires web authentication and includes AP/wireless and ICX devices, RUCKUS recommends that web authentication for wireless devices be configured directly on network AP devices and web authentication for wired devices be configured directly on the ICX devices in the network. Separating the configuration in this way prevents the dropped traffic and delays that would arise from web-authentication ports being tagged in other VLANs.
- Each Web Authentication VLAN must have a virtual interface (VE).
- The VE must have at least one assigned IPv4 address.

Web Authentication is enabled on a VLAN. That VLAN becomes a Web Authentication VLAN that does the following:

- Forwards traffic from authenticated hosts, just like a regular VLAN.
- Blocks traffic from unauthenticated hosts except from ARP, DHCP, DNS, HTTP, and HTTPs that are required to perform Web Authentication.

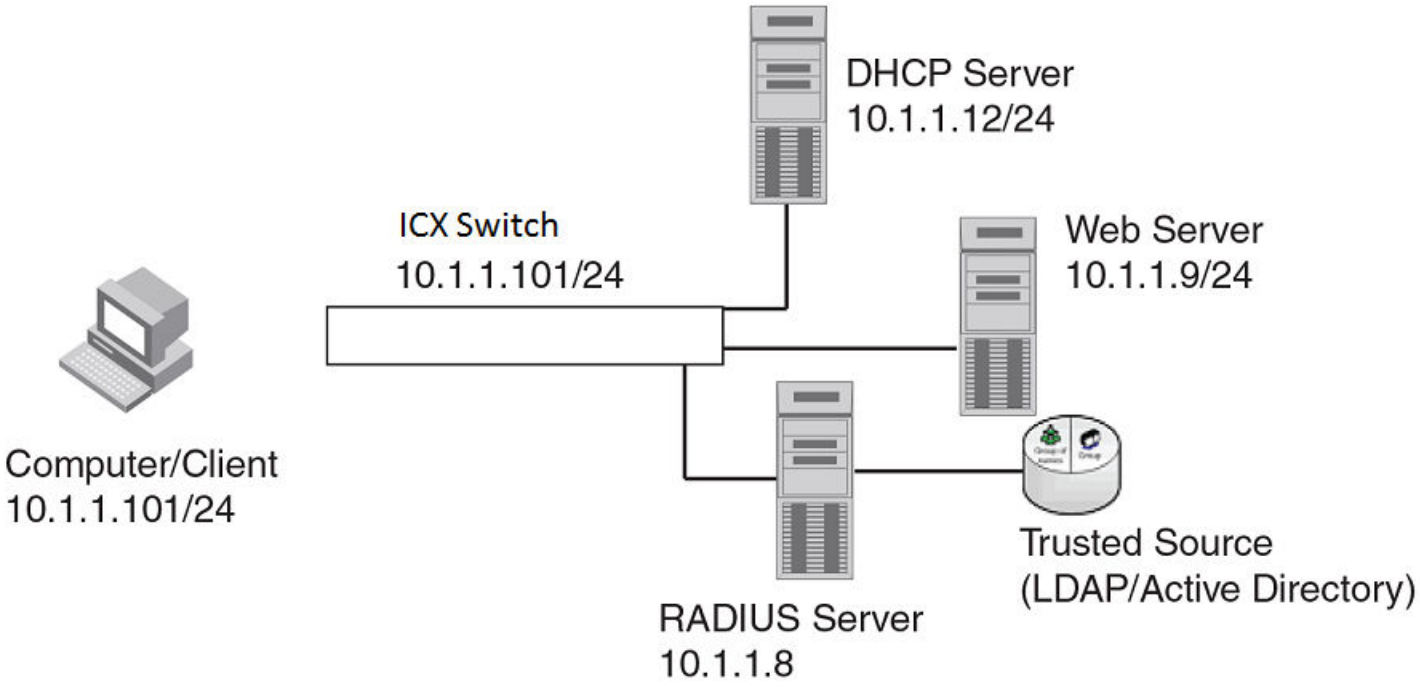
The *Basic topology for web authentication* figure shows the basic components of a network topology where Web Authentication is used. You will need:

- A RUCKUS ICX switch running a software release that supports Web Authentication
- DHCP server, if dynamic IP addressing is to be used
- Computer/host with a web browser

Your configuration may also require a RADIUS server with some Trusted Source such as LDAP or Active Directory.

NOTE
The Web server, RADIUS server, and DHCP server can all be the same server.

FIGURE 14 Basic Topology for Web Authentication



Configuring Web Authentication

Complete the following steps to configure Web Authentication on a device.

1. Enter the global configurtion mode.

```
device# configure terminal
```

2. Set up any global configuration required for the FastIron switch, RADIUS server, Web server and other servers.
   - Assign an IP address to a virtual interface (VE) for each VLAN on which Web Authentication will be enabled.

```
device(config)# vlan 10
device(config-vlan-10)# untagged e 1/1/1 to 1/1/10
device(config-vlan-10)# interface ve10
device(config-vif-10)# ip address 10.1.1.101/24
```

3. Configure the RADIUS server and other servers if Web Authentication will use a RADIUS server. By default, Web Authentication uses a RADIUS server to authenticate host usernames and passwords, unless it is configured to use a local user database.

```
device(config)# radius-server host 10.1.1.8 auth-port 1812 acct-port 1813 default key
2 $d3NpZ0BVXFpJ web-auth
```

NOTE
Remember the RADIUS key you entered. You will need this key when you configure your RADIUS server.

4. Create a Web Authentication VLAN.

```
device(config)# vlan 10
device(config-vlan-10)# webauth
```

   FastIron devices support a maximum of two Web Authentication VLANs.

5. (Optional) Configure Web Authentication to use secure (HTTPS) or non-secure (HTTP) login and logout pages.

   Web management access over HTTPS is enabled by default. For TPM-enabled devices, TPM certificates are available by default to establish encrypted communication between the server and client. For more information about digital certificates for web access, refer ICX Digital Certificates on page 63.

   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
```

6. Enable Web Authentication on the VLAN.

```
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.

7. Configure the Web Authentication mode:
   - Username and password: Blocks users from accessing the device until they enter a valid username and password on a web login page. By default "username-password" is the authentication method. For more information, refer to Using Local User Databases on page 231.
   - Passcode: Blocks users from accessing the device until they enter a valid passcode on a web login page. For more information, refer to Passcodes for User Authentication on page 232.
   - captive-portal: Authenticates the users in a VLAN through external Web Authentication (Captive Portal user authentication) mode. For more information, refer to Configuring Captive Portal (External Web Authentication) on page 228
   - None: Blocks users from accessing the device until they press the Login button. A username and password or passcode is not required. For more information, refer to Automatic Authentication on page 235.

8. Configure other Web Authentication options (refer to Web Authentication Options on page 235).

Prerequisites for Captive Portal Support with RUCKUS Cloudpath

The following are the prerequisites to support Captive Portal (external Web Authentication) on RUCKUS ICX devices.

The parameters in the following table are mandatory while creating a guest or web login page on the RUCKUS Cloudpath server.

For more details related to Web Logins page creation, refer to the *Cloudpath ES 5.2 Deployment Guide* (at this URL: https://support.ruckuswireless.com/documents/2006).

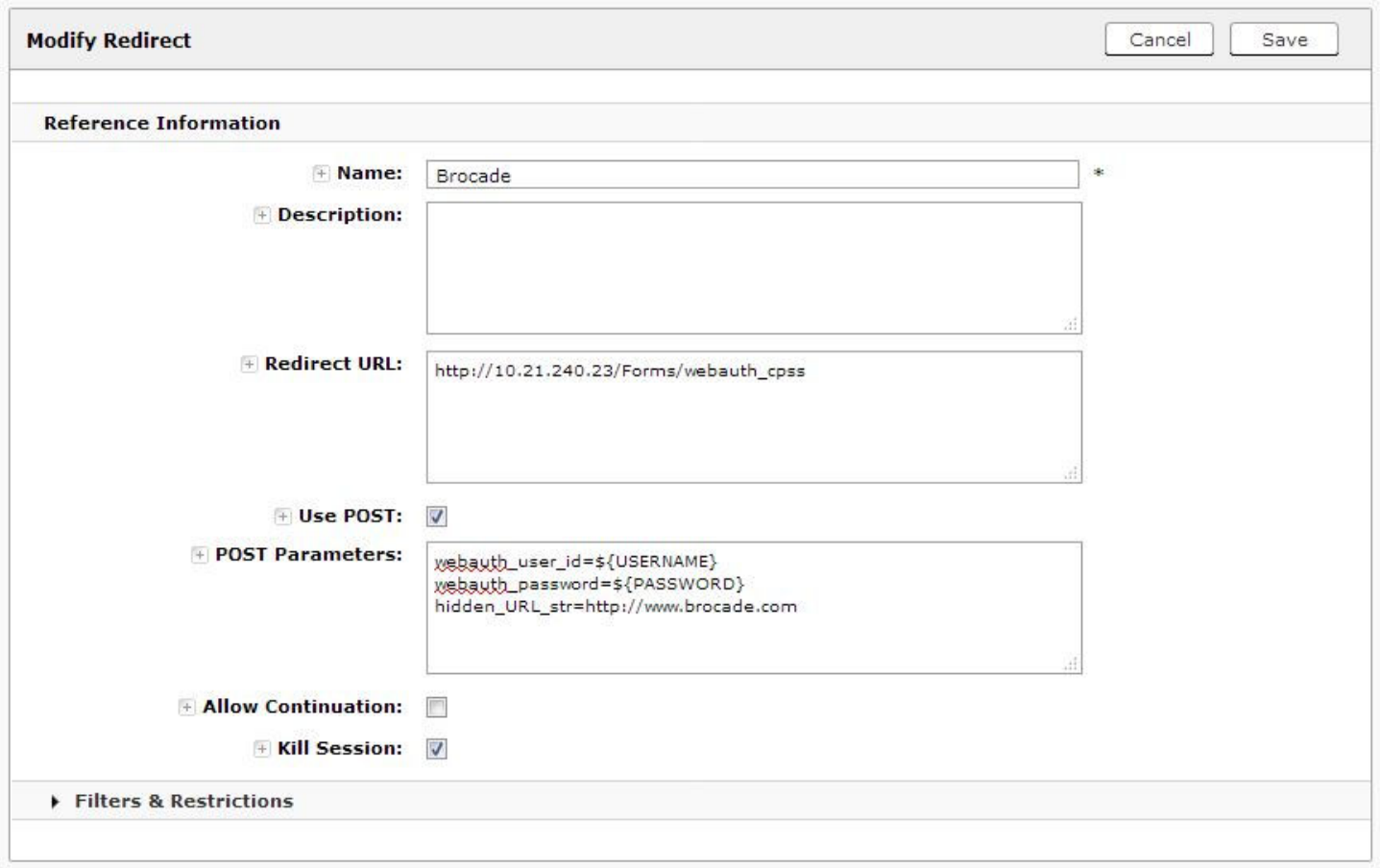
TABLE 35 Mandatory Parameters to Be Added on the RUCKUS Cloudpath Server

Fields	Value	Description
Redirect URL	Use this syntax for a single ICX switch: http://<IP address>/Forms/webauth_cpss	Specifies the URL of the NAS device's login form.
Use POST	Must be selected.	Specifies the method to use while submitting the login form to NAS.
POST Parameters	webauth_user_id=${USERNAME} webauth_password=${PASSWORD} hidden_URL_str=http://www.ruckuswireless.com	Specifies the name of the username field for the login form, the name of the password field for the login form, and the destination field for the NAS device (the default URL value).

Other vendor-specific details are selected by default.

The following figure provides an example of the information required for Web Authentication Captive Portal Redirection.

FIGURE 15 Web Login Configuration Information



Prerequisites for Configuring Captive Portal with Aruba ClearPass

The following are the prerequisites to support Captive Portal (external Web Authentication) on RUCKUS ICX devices.

- Aruba ClearPass Policy Manager or CPPM for creating and managing the security profiles used for authentication.
- Aruba ClearPass Guest module for creating web logins pages for Guest access.

The parameters in the following table are mandatory while creating a guest or web login page on the Aruba ClearPass server.

For more details related to Web Logins page creation, refer to the "Configuration" section in the Aruba ClearPass Guest User Guide, release version 6.4.

TABLE 36 Mandatory Parameters to Be Added on the Aruba ClearPass Server

Fields	Value	Description
Submit URL	Use this syntax for a single ICX switch: http://<IP address>/Forms/webauth_cpss Use this syntax for more than one ICX switches: {$switch_ip|default:"http://1.1.1.1/Forms/webauth_cpss"}	Specifies the URL of the NAS device's login form. **NOTE** The **Submit URL** value for a single ICX switch differs from the **Submit URL** value for multiple ICX switches.
Submit Method	POST	Specifies the method to use while submitting the login form to NAS.
Username Field	webauth_user_id	Specifies the name of the username field for the login form. This is passed to the NAS device when the form is submitted.
Password Field	webauth_password	Specifies the name of the password field for the login form. This is passed to the NAS device when the form is submitted.
Extra Fields	url|hidden_URL_str!=	Use this field when original client requested URL needs to be re-directed
URL Field	hidden_URL_str	Specifies the destination field for the NAS device. This field contains the default URL value.
Default URL	Any URL Example https://www.ruckuswireless.com or http://www.ruckuswireless.com	Specifies the destination URL to which the client is redirected after authentication.

Other vendor-specific details are selected by default.

The following figures show examples of the information required for Web Authentication Captive Portal Redirection.

FIGURE 16 Web Login Configuration Information

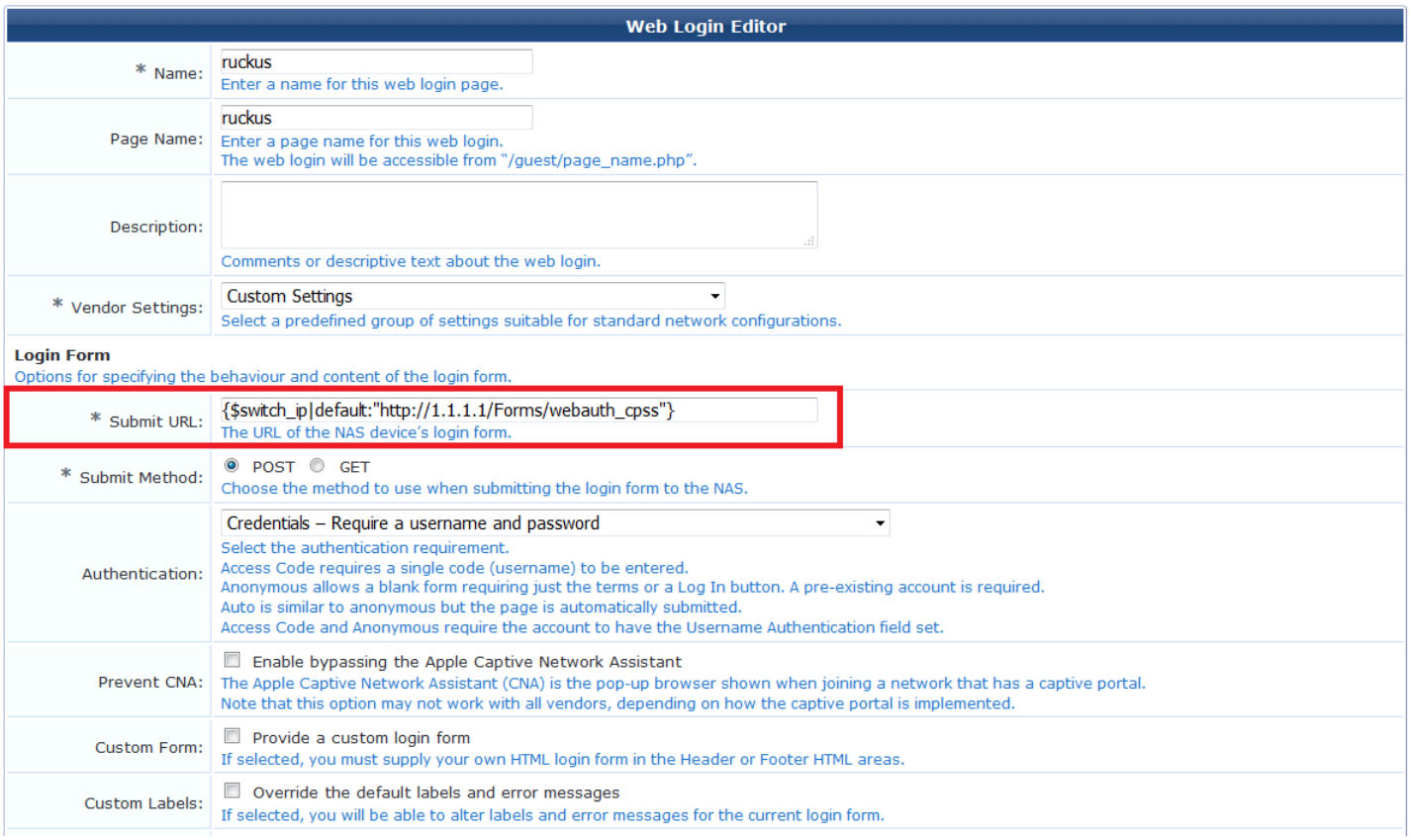
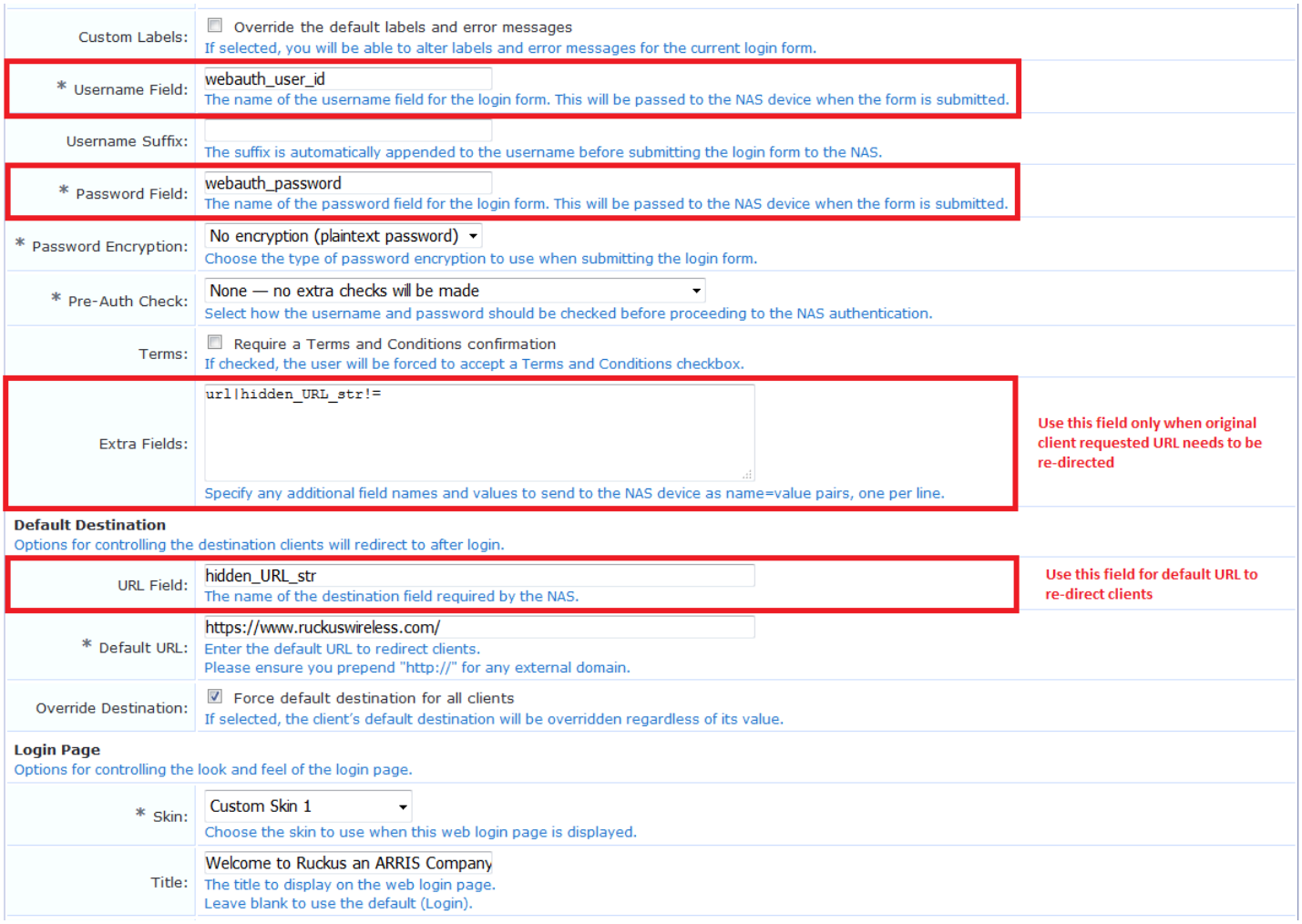


FIGURE 17 Web Login Configuration Information (cont'd)



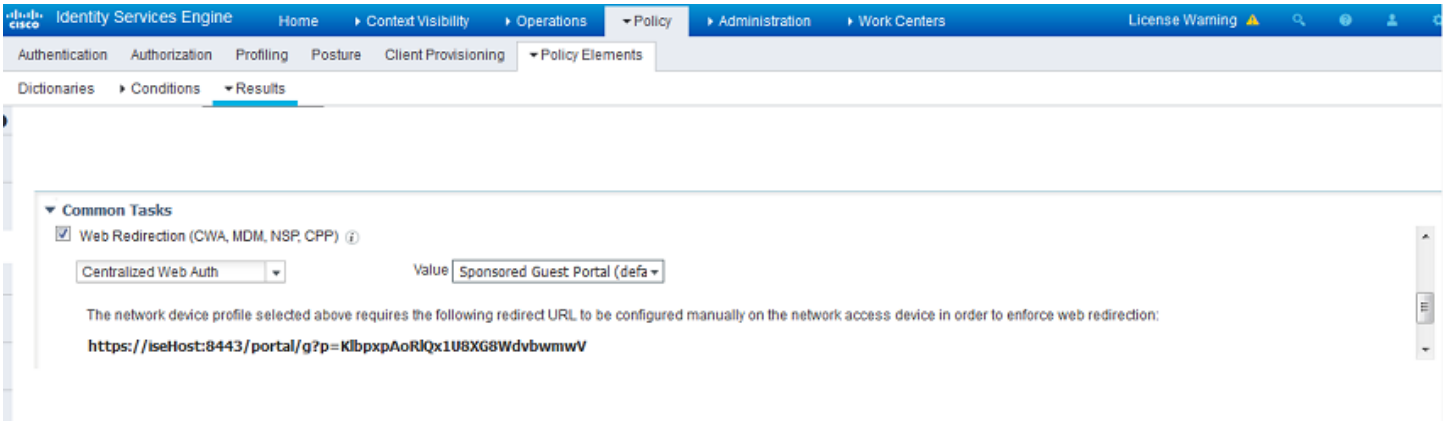
Prerequisites for Configuring External Web Authentication with Cisco ISE

External Web Authentication (also called Captive Portal) on a Cisco ISE server is supported on RUCKUS ICX devices.

Refer to the Cisco ISE server documentation for details of the redirect URL that must be configured on the RUCKUS ICX devices. To enforce web redirection, you need to manually configure the redirect URL on the ICX devices.

The following figure shows the information required for Web Authentication Captive Portal Redirection.

FIGURE 18 Cisco ISE Redirect URL



Prerequisite Configurations on an ICX Switch for Captive Portal Authentication

The following are the prerequisites to support Captive Portal (external Web Authentication) on a RUCKUS ICX device.

- Enable web management for HTTP and HTTPS access.
- Generate a crypto SSL certificate or import digital certificates issued by a third-party Certificate Authority (CA) to access a secure web page.
- Create Captive Portal profile that includes configuration details specific to the server such as virtual IP address, http or https protocol port number, and login-page details hosted on the NAC server.

Creating the Captive Portal Profile for External Web Authentication

The following steps configure the Captive Portal profile for external Web Authentication:

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

   ```
   device# configure terminal
   ```

2. Enter the **captive-portal** command to create a user-defined Captive Portal profile.

   ```
   device(config)# captive-portal cp_ruckus
   ```

   The Captive Portal command mode is enabled, where you can specify the external policy server details that enable the switch to handle the HTTP redirection mechanism.

3. Enter the **virtual-ip** command to configure the IP address of the external policy server as the virtual IP address.

   You can enter the IP address or the DNS name, which will resolve to the IP address.

   ```
   device(config-cp-cp_ruckus)# virtual-ip 10.21.240.42
   ```

4. Enter the **virtual-port** command to configure the HTTP or HTTPS protocol port number to facilitate HTTP services for the clients in external Web Authentication.

   By default, HTTPS is used and the default port number for HTTPS is 443. You can also specify HTTP mode and the default port number for HTTP is 80.

   ```
   device(config-cp-cp_ruckus)# virtual-port 80
   ```

   The protocol configured in the Captive Portal profile must be the same as the protocol configured as part of web management access using the **web-management** command.

5. Enter the **login-page** command to configure the login page details to redirect the client to the login page hosted on the external policy server. Use one of the following options, depending on which NAC server you using:

   - if you are using RUCKUS Cloudpath, use the following command: **login-page /enroll/***page-name*.

     ```
     device(config-cp-cp_ruckus)# login-page/enroll/ruckus/guestlogin
     ```

   - if you are using Aruba ClearPass, use the following command: **login-page /guest/***page-name*

     ```
     device(config-cp-cp_ruckus)# login-page/guest/ruckusguestlogin.php
     ```

   - if you are using Cisco ISE, use the following command: **login-page** *page-name created by the Cisco ISE server*.

     ```
     device(config-cp-cp_ruckus)# login-page ruckusguestlogin.php
     ```

   The login page details must be same as the login page hosted on the external policy server.

6. (Optional) Enter the **show captive-portal** command to view the output of the configured Captive Portal profile.

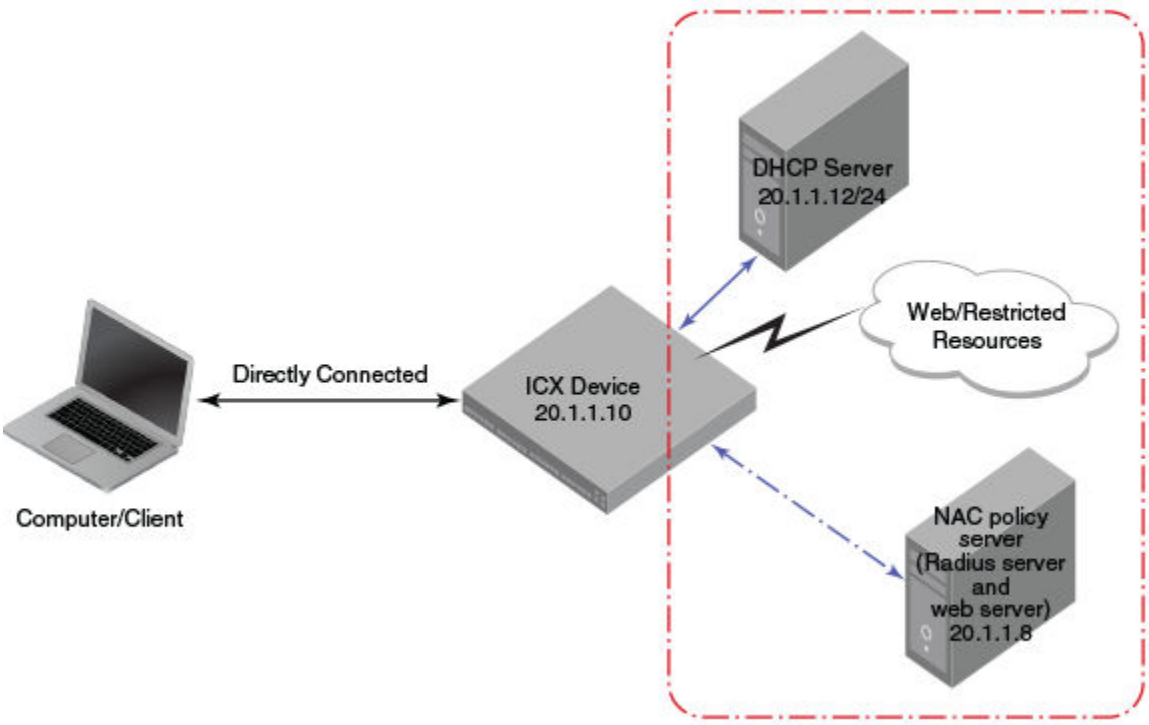
   In this example, the Captive Portal is configured with RUCKUS Cloudpath as the NAC.

   ```
   device(config)# show captive-portal cp_ruckus
   Configured Captive Portal Profile Details :
     cp-name               :cp_ruckus
     virtual-ip            :10.21.240.42
     virtual-port          :80
     login-page            :/enroll/ruckus/guestlogin.php
   ```

Configuring Captive Portal (External Web Authentication)

On ICX switches web authentication is enabled at the VLAN level. A Captive Portal profile must be applied to the web authentication-enabled VLAN. For more information, refer to Creating the Captive Portal Profile for External Web Authentication on page 227.

FIGURE 19 Basic Network Topology for a Captive Portal (External Web Authentication)



Complete the following steps to configure external Web Authentication on a device.

1. Enter the global configurtion mode.

```
device# configure terminal
```

2. Set up any global configuration required for the ICX device, RADIUS server, NAC policy server, and other servers.
   - Assign an IP address to a virtual interface (VE) for each VLAN on which external Web Authentication will be enabled.

```
device(config)# vlan 20
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
```

3. Configure the RADIUS server to authenticate the host username and passwords.

   The server has both a RADIUS server and a web server. Use the following commands to make RADIUS configuration on ICX switch.

```
device(config)# radius-server host 20.1.1.8 auth-port 1812 acct-port 1813 default key
2 $d3NpZ0BVXFpJ web-auth
```

   NOTE

   The RADIUS key configured should be the same as the key configured in the external web server.

4. Create a Web Authentication VLAN.

```
device(config)# vlan 20
device(config-vlan-20)# webauth
```

5. (Optional) Configure Web Authentication to use secure (HTTPS) or non-secure (HTTP) login and logout pages.

   Web management access over HTTPS is enabled by default. For TPM-enabled devices, TPM certificates are available by default to establish encrypted communication between the server and client. For more information about digital certificates for web access, refer ICX Digital Certificates on page 63.

   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
   ```

6. Enable Web Authentication on the VLAN.

   ```
   device(config-vlan-20-webauth)# enable
   ```

   From this step onwards, the hosts must be authenticated to forward traffic.

7. Attach the configured Captive Portal profile to the Web Authentication-enabled VLAN.

   ```
   device(config-vlan-20-webauth)# captive-portal profile cp_ruckus
   ```

8. 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
   ```

9. Configure the external Captive Portal on the NAC server to create a guest or web login page for external Web Authentication.

Web Authentication Mode Configuration

You can configure the FastIron device to use one of four Web Authentication modes:

- Username and password: Blocks users from accessing the device until they enter a valid username and password on a web login page. By default "username-password" is the authentication method. For more information, refer to Using Local User Databases on page 231.
- Passcode: Blocks users from accessing the device until they enter a valid passcode on a web login page. For more information, refer to Passcodes for User Authentication on page 232.
- Captive Portal: Authenticates the users in a VLAN through external Web Authentication (Captive Portal user authentication) mode.
- None: Blocks users from accessing the device until they press the Login button. A username and password or passcode is not required. For more information, refer to Automatic Authentication on page 235.

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 and selects username-password as the web authentication mode.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Create a local user database.

```
device(config)# local-userdb userdb1
```

   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.

3. Add user records to the local user database using one of the following methods:

   - Enter a user record that consists of username and password.

```
device(config-localuserdb-userdb1)# username marcia password bunch4
```

   - Import ASCII text file of user records from the TFTP server.

```
device(config-localuserdb-userdb1)# import-users tftp 192.168.1.1 filename userdb1
```

   The **no username** command deletes a user record from the local user database. To delete all user records, enter the **delete-all** command.

   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.

4. Enable the username-password authentication mode.

```
device(config-vlan-10-webauth)# auth-mode username-password
```

   Complete the configuration steps described in Configuring Web Authentication on page 221 before this configuration.

5. Set one of the following authentication methods.

   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.

   - Configure the switch to use a local user database to authenticate users in a VLAN.

```
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
```

6. If desired, set the authentication method (RADIUS or local) failover sequence.

```
device(config-vlan-10-webauth)# auth-mode username-password auth-methods local radius
```

   You can specify a failover sequence for the RADIUS and local user database authentication methods. In this example Web Authentication is configured 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.You can specify **radius local** or **local radius** depending on the failover sequence desired.

7. Assign a local user databse to a Web Authentication VLAN.

   After creating or importing a local user database on the device 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.

```
device(config-vlan-10-webauth)# auth-mode username-password local-user-database userdb1
```

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.

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.

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.

Passcode Refresh Methods

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.

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.

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.

> **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.

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.

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.

Manual Passcode Refresh

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.

Configuring Passcode Authentication

The following steps configure the device to use the passcode authentication mode.

1. Complete the configuration steps described in Configuring Web Authentication on page 221.
2. Create static passcodes.

```
device(config-vlan-10-webauth)# auth-mode passcode static 56127
```

3. Enable passcode authentication.

```
device(config-vlan-10-webauth)# auth-mode passcode
```

   This 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.

4. (Optional) Configure the length of dynamically generated passcodes.

```
device(config-vlan-10-webauth)# auth-mode passcode length 10
```

   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.

5. (Optional) Configure one of the following passcode referesh methods:
   - Configure the duration of time (in minutes) after which passcodes must be refreshed.

```
device(config-vlan-10-webauth)# auth-mode passcode refresh-type duration 4320
```

   - Configure the time of day at which the passcodes must be refreshed.

```
device(config-vlan-10-webauth)# auth-mode passcode refresh-type time 6:00
```

   By default, passcodes will be refreshed at 00:00 (12:00 midnight). You can configure up to 24 refresh periods in a 24-hour period. Each must be at least five minutes apart.

   a) (Optional) Reset the passcode refresh time of day configuration 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
```

6. (Optional) Configure a grace period for an expired passcode.

```
device(config-vlan-10-webauth)# auth-mode passcode grace-period 5
```

7. (Optional) Delete all expired passcodes that are currently in the grace period.

```
device(config-vlan-10-webauth)# auth-mode passcode flush-expired
```

8. (Optional) Disable and re-enable passcode log.

   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.

```
device(config-vlan-10-webauth)# no auth-mode passcode log syslog
device(config-vlan-10-webauth)# auth-mode passcode log syslog
device(config-vlan-10-webauth)# no auth-mode passcode log snmp-trap
device(config-vlan-10-webauth)# auth-mode passcode log snmp-trap
```

9. (Optional) Retransmit the current passcode to a Syslog message or SNMP trap.

```
device(config-vlan-10-webauth)# auth-mode passcode resend-log
```

   The switch retransmits the current passcode only. Passcodes that are in the grace period are not sent.

10. (Optional) Manually refresh the passcode.

```
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 Permanently Authenticated Hosts on page 236.) You must still specify devices that are to be permanently authenticated even if automatic authentication is enabled.

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.

Configuring Automatic Authentication

The following steps configure the Web Authentication mode to authenticate the hosts automatically.

1. Complete the configuration steps described in Configuring Web Authentication on page 221.
2. Configure the Web Authentication mode to enable automatic authentication.

```
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.

Web Authentication Options

Web Authentication also offers the following configuration options:

- RADIUS accounting
- Trusted ports
- Permanently authenticated hosts
- White-lists
- Re-authentication period
- Web authentication time limit
- Maximum web authentication attempts
- Clearing of authenticated hosts from the Web Authentication table

- Block duration for too many authentication attempts
- Manual blocking of a specific host
- Limiting the number of authenticated hosts
- Filtering DNS queries
- Forced re-authentication
- Web authorization redirect and redirect page authentication
- Honoring the RADIUS-returned VLAN

RADIUS Accounting for Web Authentication

When Web Authentication is enabled, you can enable RADIUS accounting using the **accounting** command to record login (start) and logout (stop) events per host. The information is sent to a RADIUS server. Note that packet and byte count is not supported.

Trusted Ports

You can configure certain ports of a Web Authentication VLAN as trusted ports using the **trust-port** command. All hosts connected to the trusted ports need not authenticate and are automatically allowed access to the network.

Permanently Authenticated Hosts

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. Some of these hosts (such as printers) may not have a web browser and will not be able to perform Web Authentication. Such hosts can be permanently authenticated using the **add mac** command. You can set the duration to specify how long the MAC address remains authenticated. The default is the time configured using the **reauth-time** command. To keep the host permanently authenticated, set the **duration** to **0** so that the Web Authentication for the MAC address does not expire.

Instead of simply 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.

NOTE
If a MAC address is statically configured, it will not be dynamically configured on any port.

IP Addresses and Domain Names Allowed During Web Authentication

You can create a list of preconfigured hosts and servers that are allowed access during Web Authentication. This capability is typically referred to as a walled garden. Depending on the Web Authentication processes followed by users, communication with more than one host or server may be required during authentication. A typical example would be when Web Authentication requires certificate exchanges with a specific host.

Web Authentication automatically allows DHCP or DNS packets, depending on the protocol used, and allows access to the Captive Portal server used for authentication. Any other sites that may be required can be configured as white-lists at the VLAN level. An IPv4 address with or without a subnet mask or a fully qualified domain name (FQDN) can be configured as a Web Authentication white-list. Up to 100 white-lists can be configured for Web Authentication.

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. The number of seconds a host remains authenticated before being logged off can be configured using the **reauth-time** command.

Web Authentication Cycle

You can specify the amount of time allowed for a user to authenticate successfully, starting from the first Login attempt on the Login page. When the time expires, the user must enter a valid URL again to display the Web Authentication welcome page.

Limiting the Number of Web Authentication Attempts

You can use the **attempt-max-num** command to limit the number of times a user enters an invalid username and password during the Web Authentication cycle. If the user exceeds the limit, the user is blocked for the time defined by the **block duration** command and is redirected to the exceeded allowable attempts web page.

Clearing Authenticated Hosts from the Web Authentication Table

You can clear dynamically authenticated hosts from the Web Authentication table. All authenticated hosts in a Web Authentication VLAN can be cleared using the **clear webauth vlan** *vlan-id* **authenticated-mac** command. If you want to clear a particular host in a Web Authentication VLAN, use the **clear webauth vlan** *vlan-id* **authenticated-mac** *mac-address* command.

Block Duration for Web Authentication Attempts

You can use the **block duration** command to specify how long users must wait to try again after exceeding the number of allowed Web Authentication attempts.

To unblock the MAC address, wait until the block duration timer expires, or enter the **clear webauth vlan** *vlan-id* **block-mac** *mac-address* command.

Manually Blocking a Specific Host

A host can be temporarily or permanently blocked from attempting Web Authentication **block mac** *mac-address* **duration** *time* command.

You can specify the duration from 0 through 128000 seconds. The default is the current value of the **block duration** command. To keep the MAC address permanently blocked, set the **block mac** *mac-address* **duration** to 0.

Limiting the Number of Authenticated Hosts

You can limit the number of hosts that are authenticated at a time by entering the **host-max-num** command. You can specify from 0 through 8192 hosts. The default value is 0, which means that 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 ICX 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 using the **dns-filter** command. 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 and specify a number from 1 through 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.

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.

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 seconds.

Defining the Web Authorization Redirect Address

When a user enters a valid URL, the user is redirected to the Web Authentication welcom page. By default, the Web Authentication address returned to the browser is the IP address of the ICX 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 using the **webauth-redirect-address** command.

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.

Customizing the Web Authentication Redirect Page

The Web Authentication login page presented by ICX devices (except when the external captive portal is used) contains the prompts for authentication credentials. These prompts are labeled **UserId** and **Password** by default. The administrator can remove the **UserId** label and associated entry field from the webpage. The administrator can also rename the **UserId** and **Password** labels on the authentication page.

Honoring the RADIUS-Returned VLAN

NOTE
Honoring the RADIUS-returned VLAN applies to network segmentation deployments only.

If the RADIUS server returns a VLAN attribute when client authentication is successful, the client's connected access port is moved to the RADIUS-returned VLAN, and the uplink port is also added to the RADIUS-returned VLAN. The resulting configuration is saved. Any clients that subsequently connect to the uplink port will be allowed access.

Configuring Web Authentication Options

The following steps configure the options available for Web Authentication.

1. Complete the configuration steps described in Configuring Web Authentication on page 221.
2. Enable RADIUS accounting.

```
device(config-vlan-10-webauth)# accounting
```

3. Create a list of trusted ports.

```
device(config-vlan-10-webauth)# trust-port ethernet 1/1/3
device(config-vlan-10-webauth)# trust-port ethernet 1/1/6 to 1/1/10
```

4. Configure a host to be permanently authenticated.

```
device(config-vlan-10-webauth)# add mac 0000.00eb.2d14 duration 0
```

5. Configure any additional sites that must be allowed Web Authentication access, for example, for certification. Enter the **white-list** command followed by an ID and an IPv4 address or the FQDN of the site. The ID can be a decimal value from 1 through 100. Create additional white-lists if needed, using one address or FQDN per command.

   NOTE
   The IP address is resolved against the FQDN at the time of configuration only. The ICX device does not attempt to resolve the IP address during client authentication.

```
device(config-vlan-10-webauth)# white-list 10 www.commscope.com
```

   The previous example adds an FQDN (www.commscope.com) to the servers and sites allowed access during Web Authentication on the ICX device.

```
device(config-vlan-10-webauth)# white-list 30 192.168.0.1/24
```

   The previous example configures an IPv4 subnet as a Web Authentication white-list.

```
device(config-vlan-10-webauth)# white-list 35 192.168.12.2 255.255.255.0
```

   The previous example configures an IPv4 server address as a Web Authentication white-list.

6. Configure the re-authentication period (number of seconds a host remains authenticated before being logged off).

```
device(config-vlan-10-webauth)# reauth-time 10
```

   You can specify 0 through 128000 seconds. The default is 28800 seconds, and 0 means the host 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 host becomes de-authenticated if there is no activity and the timer for the inactive period expires.

7. Define the authentication cycle time.

```
device(config-vlan-10-webauth)# cycle time 20
```

8. Limit the number of Web Authentication attempts.

```
device(config-vlan-10-webauth)# attempt-max-num 4
```

9. Set the block duration for Web Authentication attempts, specifying how many seconds users must wait before the next cycle of Web Authentication begins.

```
device(config-vlan-10-webauth)# block duration 4
```

10. Manually block a specific host from attempting Web Authentication.

```
device(config-vlan-10-webauth)# block mac 0000.00d1.0a3d duration 4
```

11. Limit the number of hosts that can be authenticated at a time.

```
device(config-vlan-10-webauth)# host-max-num 300
```

12. Define Domain Name System (DNS) filters that will restrict DNS queries from unauthenticated hosts to be forwarded explicitly to defined

```
device(config-vlan-10-webauth)# dns-filter 1 10.166.2.44/24
```

13. Enable forced reauthentication of the hosts if all the ports on the device go down.

```
device(config-vlan-10-webauth)# port-down-authenticated-mac-cleanup
```

14. Configure the time duration after which the user-associated MAC address is aged out and reauthentication is enforced.

```
device(config-vlan-10-webauth)# authenticated-mac-age-time 300
```

15. Configure a redirect address for Web Authentication to prevent the display certificate mismatch error message.

```
device(config-vlan-10-webauth)# webauth-redirect-address my.domain.net
```

16. Configure an uplink port or LAG to be used as the connection to your uplink switch.

NOTE
The client for which the uplink port is configured must not be a member of the Flexible authentication VLAN.

```
device(config-vlan-10-webauth)# uplink-port ethernet 1/1/1
device(config-vlan-10-webauth)# uplink-port lag 2
```

17. Clear authenticated hosts from the Web Authentication table.

```
device# clear webauth vlan 25 authenticated-mac
```

Web Authentication Pages

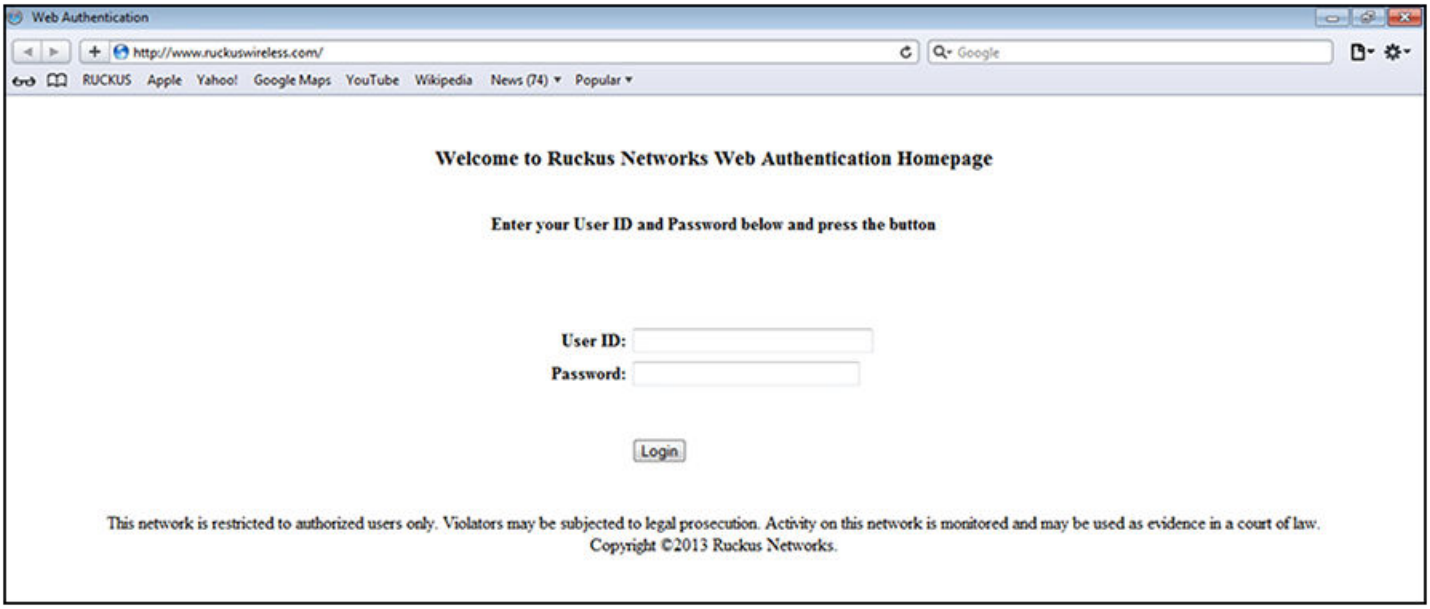
Several different web pages may be displayed during Web Authentication.

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 238).

If automatic authentication is enabled, a welcome page is displayed. The browser will then be directed to the requested URL.

If username and password (local user database) authentication is enabled, the following login page is displayed by default.

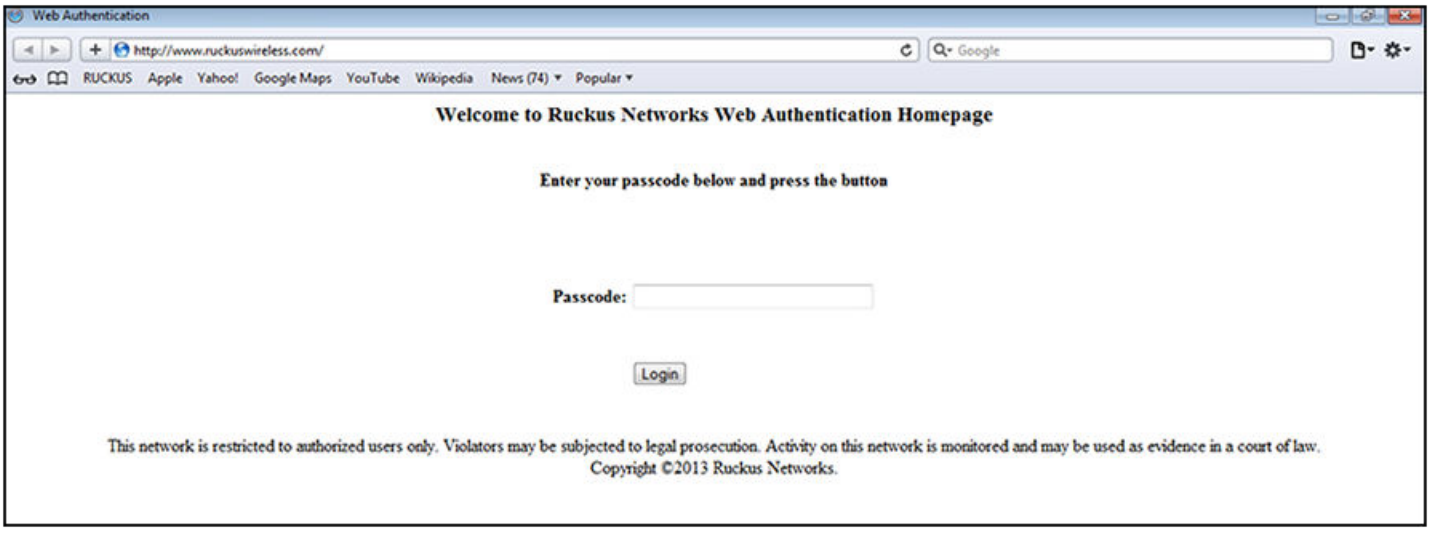
FIGURE 20 Example of a Default Login Page



The user enters a username and password, which are sent for authentication.

If passcode authentication is enabled, the following login page is displayed.

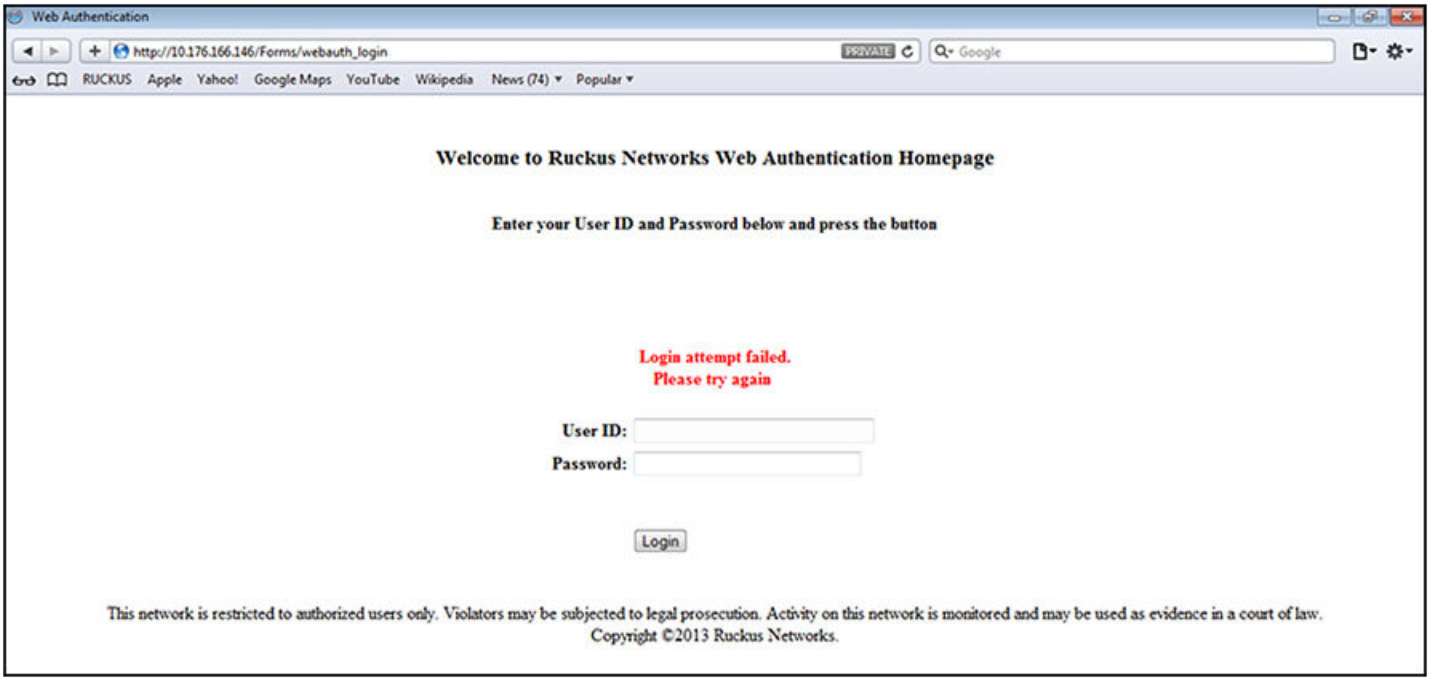
FIGURE 21 Example of Login Page When Automatic Authentication Is Disabled and Passcode Authentication Is Enabled



The user enters a passcode, which is sent for authentication.

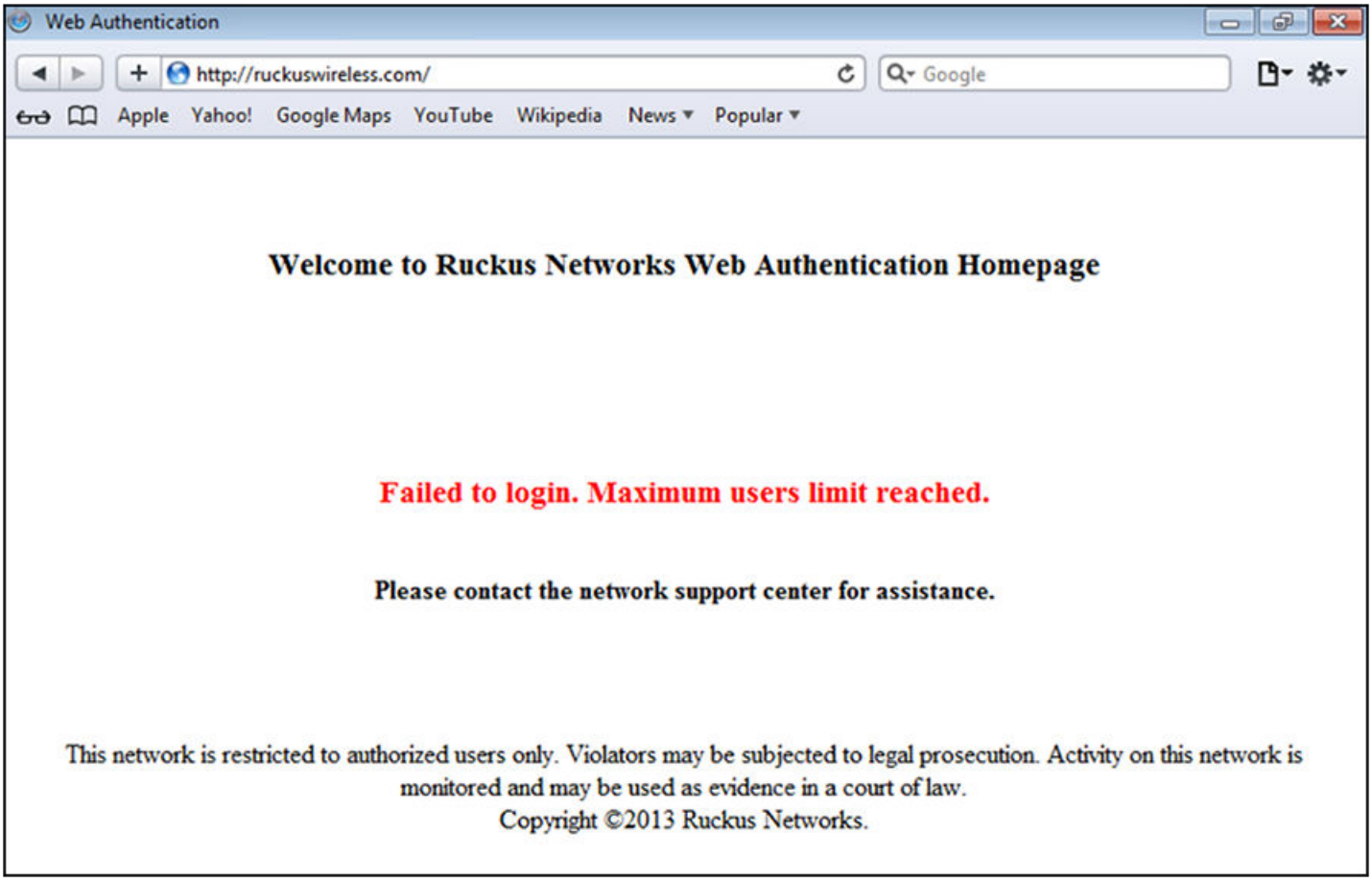
If Web Authentication fails, the following "try again" page is displayed.

FIGURE 22 Example of a Try Again Page



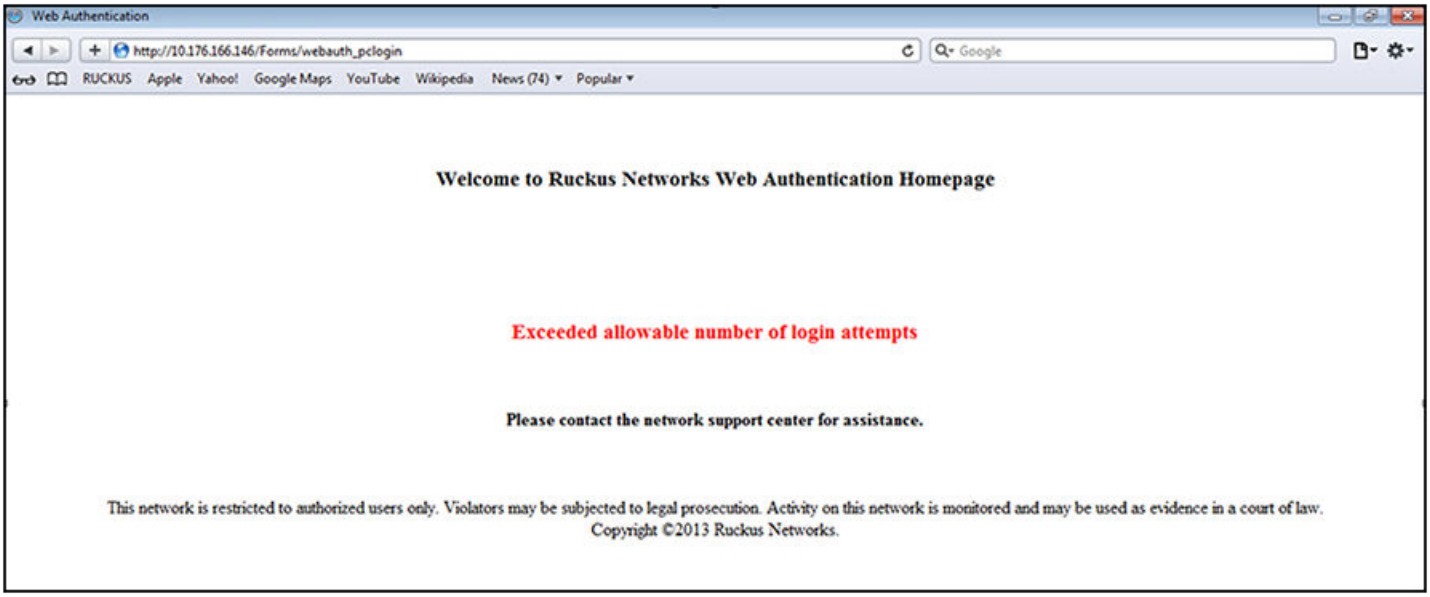
If the limit for the number of authenticated users on the network is exceeded, the following maximum host limit page is displayed.

FIGURE 23 Example of a Maximum Host Limit Page



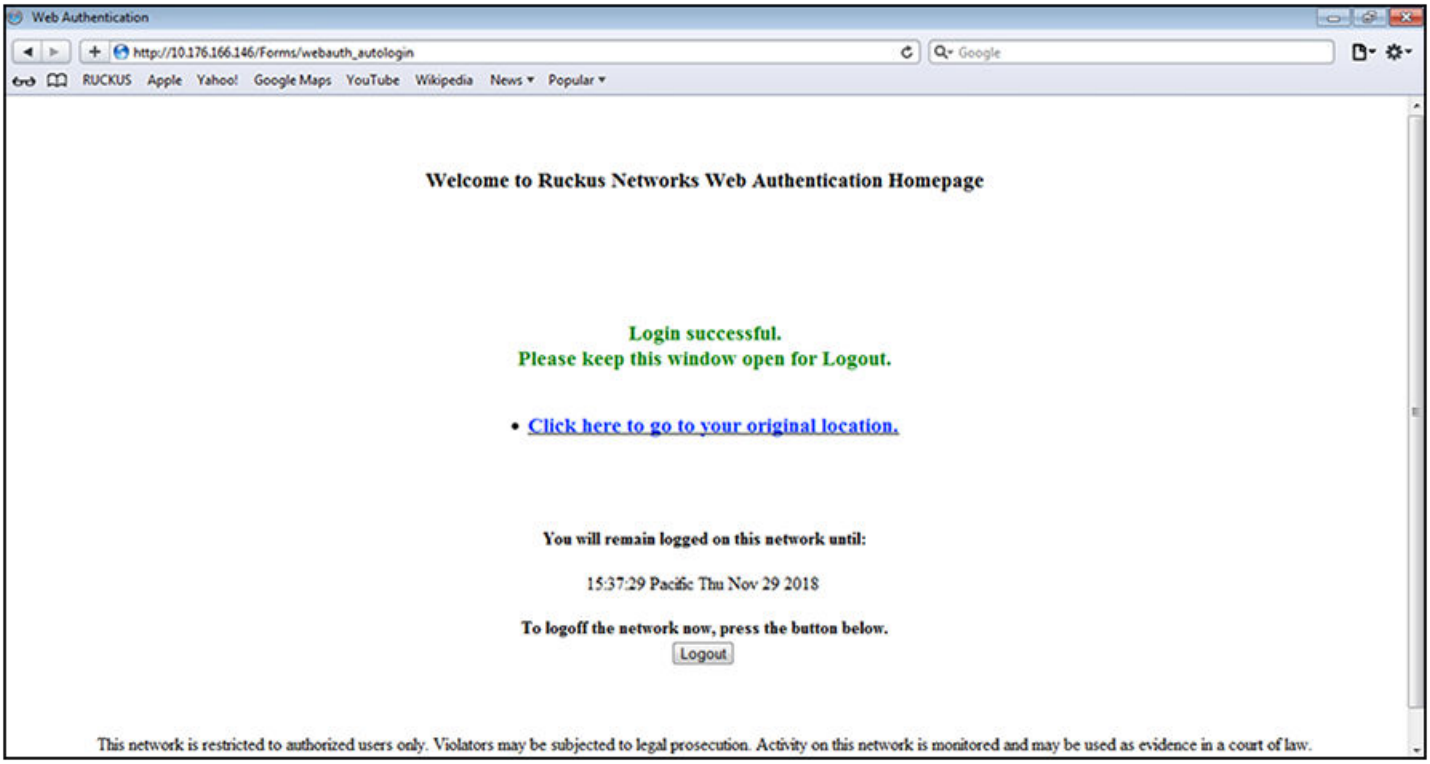
If the number of Web Authentication attempts by a user has been exceeded, the maximum attempts limit page is displayed. The user is blocked from attempting Web Authentication until either the user MAC address is removed from the blocked list (using the **clear webauth block-mac** command) or the block duration timer expires.

FIGURE 24 Example of a Maximum Attempts Limit Page



If Web Authentication is successful, the following success page is displayed.

FIGURE 25 Example of a Web Authentication Success Page

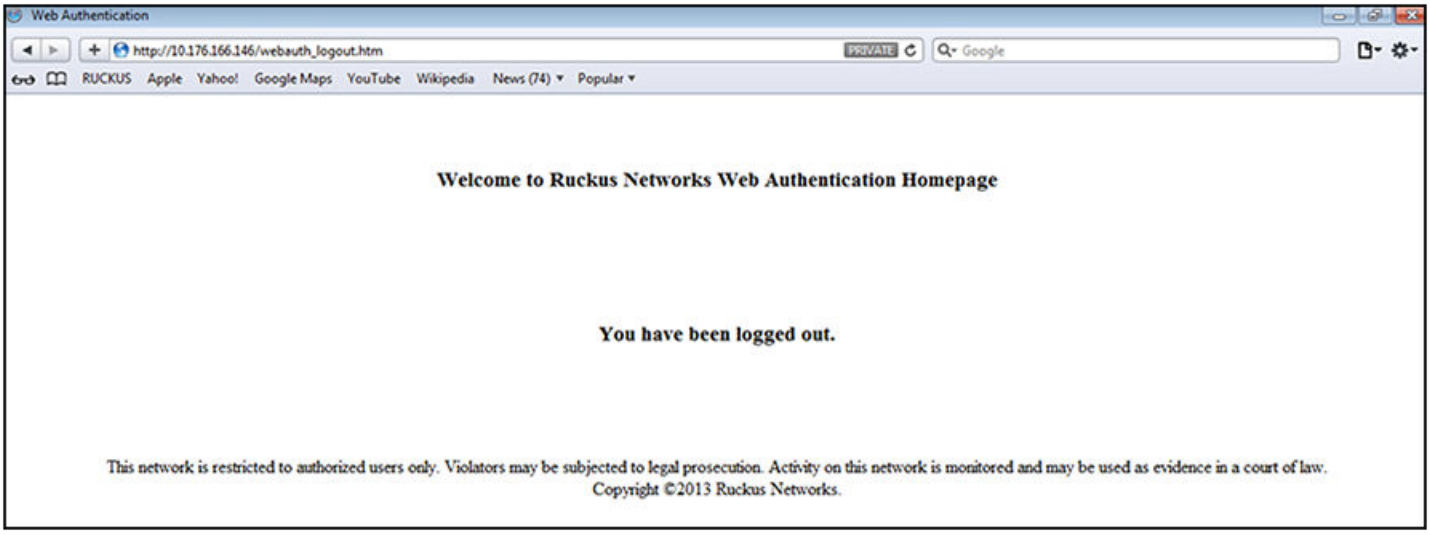


Once a host is authenticated, that host can manually de-authenticate by clicking the **Logout** button on the login success page. The host remains logged in until the re-authentication period expires. At that time, the host is automatically logged out. However, if a re-authentication period is not configured, 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.

You can log out of the host session by clicking the **Logout** button. Once the host is logged out, the following logout message page is displayed.

FIGURE 26 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.

Customizing Web Authentication Pages

You can customize the following objects in the Web Authentication pages:

- Title bar
- Banner image (logo)
- Header
- Textbox
- **Login** button
- **User ID** label and entry field
- **Password** label and entry field
- Footer

You can use the **show webauth** or the **show webauth vlan** *vlan-ID* **webpage** command 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 entry fields and **Login** button apply to the login page only.

The following steps customize the objects in the Web Authentication page.

1. Customize the title bar that is displayed on all Web Authentication pages.

```
device(config-vlan-10-webauth)# webpage custom-text title "Ruckus Secure Access Page"
```

2. Customize the banner image (logo) using TFTP.

```
device(config-vlan-10-webauth)# webpage logo copy tftp 10.10.5.1 ruckuslogo.gif
```

   To customize the banner image, use the TFTP protocol to upload an image file from a TFTP server to the ICX device. The image file can be in the .jpg, .bmp, or .gif format, and must be 64 Kb or less. If you upload a new image file, it will overwrite the existing image file.

   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.

3. Align the banner image (logo).

```
device(config-vlan-10-webauth)# webpage logo align center
```

   By default, the logo is left-aligned at the top of the page. You can change the placement of the logo to the center or right at the top of the page.

4. Customize the header.

```
device(config-vlan-10-webauth)# webpage custom-text top "Welcome to Network One"
```

5. Customize the text box.

   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 ICX device. The text file size must not exceed 2 Kb.

```
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.

6. Customize the **Login** button.

```
device(config-vlan-10-webauth)# webpage custom-text login-button "Press to Log In"
```

   The default **Login** button text is "Login".

7. Remove or customize the **User ID** field.

   NOTE
   Before you use the **remove-userid-label**, be sure the user ID and password match those stored on the authentication server.

   The following example removes the **User ID** field from the web page.

```
device(config-vlan-10-webauth)# webpage remove-userid-label
```

   The following example changes the **User ID** label to read "username". The alphanumeric text used for the label has a maximum length of 32 characters.

```
device(config-vlan-10-webauth)# webpage custom-label username-label "username"
```

8. Customize the **Password** field.

   The following example changes the **Password** label to read "password to authenticate". The alphanumeric text used for the label has a maximum length of 32 characters.

   ```
   device(config-vlan-10-webauth)# webpage custom-label password-label "password to authenticate"
   ```

9. Customize the footer.

   ```
   device(config-vlan-10-webauth)# webpage custom-text bottom "Network One Copyright 2021"
   ```

Displaying Web Authentication Information

You can use a number of **show** commands to display information about Web Authentication.

The following example displays the Web Authentication configuration details.

```
device(config-vlan-300-webauth)# show webauth

==========================================================================
WEB AUTHENTICATION (VLAN 300): 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)
webauth-redirect-address: none (Default)
dns-filter: Disable (Default)
white-list: Disable (Default)
authentication mode: username and password (Default)
  authentication methods: radius
    Local user database name:  <none>
Radius accounting: Enable (Default)
Trusted port list: lag 1
Uplink port : lag 1
Secure Login (HTTPS): Disable
Web Page Customizations:
  Top (Header): Custom Text
        "Test"
  Bottom (Footer): Default Text
  Title: Default Text
  Login Button: Custom Text
        "Press Login"
  Web Page Logo: None (Default)
    align: left (Default)
  Web Page Terms and Conditions: None (Default)
Web Page UserId label: username
Web Page Password label: password to authenticate
Host statistics:
 Number of hosts dynamically authenticated: 0
 Number of hosts statically authenticated: 0
 Number of hosts dynamically blocked: 0
 Number of hosts statically blocked: 0
 Number of hosts authenticating: 0
```

The **show webauth** command by itself displays information for all VLANs on which Web Authentication is enabled. The **show webauth vlan** *vlan-id* command displays information for a specific VLAN.

The following example displays the Web Authentication configuration details for a specific VLAN.

```
device(config-vlan-300-webauth)# show webauth vlan 300

==========================================================================
```

```
WEB AUTHENTICATION (VLAN 300): 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)
webauth-redirect-address: none (Default)
dns-filter: Disable (Default)
white-list: Disable (Default)
authentication mode: username and password (Default)
  authentication methods: radius
    Local user database name:  <none>
Radius accounting: Enable (Default)
Trusted port list: lag 1
Uplink port : lag 1
Secure Login (HTTPS): Disable
Web Page Customizations:
  Top (Header): Custom Text
        "Test"
  Bottom (Footer): Default Text
  Title: Default Text
  Login Button: Custom Text
        "Press Login"
  Web Page Logo: None (Default)
    align: left (Default)
  Web Page Terms and Conditions: None (Default)
Web Page UserId label: username
Web Page Password label: password to authenticate
Host statistics:
 Number of hosts dynamically authenticated: 0
 Number of hosts statically authenticated: 0
 Number of hosts dynamically blocked: 0
 Number of hosts statically blocked: 0
 Number of hosts authenticating: 0
```

The following example displays the text that has been configured for Web Authentication pages.

```
device(config-vlan-300-webauth)# show webauth vlan 300 webpage

================================
Web Page Customizations (VLAN 300):
  Top (Header): Default Text
        "<h3>Welcome to Ruckus Networks Web Authentication Homepage</h3>"
  Bottom (Footer): Default Text
        "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.<br>Copyright &copy;2013 Ruckus Networks."
  Title: Default Text
        "Web Authentication"
  Login Button: Default Text
        "Login"
  Web Page Logo: None (Default)
    align: left (Default)
  Web Page Terms and Conditions: None (Default)
Web Page UserId label: username
Web Page Password label: password to authenticate
Web Page UserId label not displaying
ICX7650-48P Router(config-vlan-300-webauth)#
```

The following example displays a list of hosts that are currently authenticated.

```
device# show webauth allowed-list
================================================================================
VLAN 3: Web Authentication, Mode: I = Internal E = External
--------------------------------------------------------------------------------
Web Authenticated List               Configuration   Authenticated Duration  Dynamic
MAC Address       User Name   mode    Static/Dynamic  HH:MM:SS                ACL
--------------------------------------------------------------------------------
000c.2973.a42b    ruckus      E       D               1 day, 11:33:16         acl1
```

```
1222.0a15.f045    super       E       D              1 day, 11:32:51         acl1
1222.0a15.f044    foundry     E       D              1 day, 11:32:48         acl1
1222.0a15.f043    ruckus      E       D              1 day, 11:32:47         acl1
1222.0a15.f042    spirent     E       D              1 day, 11:32:4          acl1
```

The following example displays 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 following example displays a list of hosts that are currently blocked from any Web Authentication attempt.

```
device# show webauth blocked-list
==============================================================================
VLAN 3: Web Authentication, AuthMode: I=Internal E=External
------------------------------------------------------------------------------
Block List                        Configuration       Block Duration Remaining
MAC Address     User Name  Mode   Static/Dynamic      HH:MM:SS
------------------------------------------------------------------------------
000c.2973.a42b  User1      E      D                   00:00:04
```

The following example displays a list of all local user databases configured on the ICX 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
```

The following example 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.

The following example displays current passcodes if the passcode Web Authentication mode is enabled.

```
device# show webauth vlan 25 passcode
Current Passcode : 1389
This passcode is valid for 35089 seconds
```

The following example displays the details of the captive portal profile configured on the device.

```
device(config)# show captive-portal cp-ruckus
Configured Captive Portal Profile Details :
  cp-name              :cp-ruckus
  virtual-ip           :10.21.240.42
  virtual-port         :80
  login-page           :/enroll/ruckus/guestlogin
```

Image Download over HTTPS

Support has been added for downloading images to flash memory over HTTPS. HTTPS image download involves downloading of an image from the remote HTTPS server to either the primary or secondary partition in a simple process.

After the **copy https flash** command is executed specifying the URL to the image file on the server (including IP address, port, path, and file name), an HTTP request is issued to download an image from the server. The response to the HTTP request contains the binary image in the body. After the image is downloaded by the HTTPS client, the data is written to a file in a specific location and image installation begins for either the primary or secondary partition.

Considerations for Downloading Images over HTTPS

- The flash memory is locked for the entire image download and installation process.
- Primary and secondary images are supported.
- The HTTPS client uses the device certificate when it is connected to the a SZ server. In other cases it can connect to a server, for example a Linux HTTPS server, without a device certificate.
- The remote end must serve the ICX image, either by a web server or have a web service providing the image for the URL specified in the HTTP request.
- No special HTTP headers are required.
- Image download over HTTPS is not available in FIPS or CC mode.
- Only IPv4 HTTPS download is supported.
- A successful HTTPS download provides a 200 response code.
- If a unified forwarding image (UFI) is specified, the UFI consists of the application image, the boot code image and the signature in one unified file. For more information on the UFI, refer to the *Software Upgrade and Downgrade* chapter in the *RUCKUS FastIron Software Upgrade Guide*.

Downloading an Image from a Web Server to Flash Memory

The following example copies the “SPR08070b1.bin” image from the HTTPS server to the flash primary partition. IP address 10.1.1.1 is specified and port 876 is specified.

```
device# copy https flash 10.1.1.1 SPR08070b1.bin primary port 876
```

The following example copies the “SPR08070b1.bin” image from the HTTPS server to the flash secondary partition. IP address 10.2.1.1 is specified. Because no port is specified, the default of 443 is used.

```
device# copy https flash 10.2.1.1 SPR08070b1.bin secondary
```

The following example copies a primary UFI image file from the HTTPS server to the flash primary partition. IP address 10.2.1.1 is specified and port 700 is specified. The UFI consists of the application image, the boot code image, and the FI signature in one unified file.

```
device# copy https flash 10.2.1.1 SPR08080b1ufi.bin primary port 700
```

Configuration Download over HTTPS

Support has been added for downloading a configuration file from the HTTPS server to the startup configuration file. HTTPS configuration download works in a simple process.

After the **https startup-config** command is executed specifying the URL to the file on the server (including IP address, port, path, and file name), an HTTP request is issued to download the configuration file from the server. The response to the HTTP request contains the configuration in the body. After the configuration is downloaded by the HTTPS client, the data returned is written to the startup configuration file, replacing the original startup configuration.

Considerations for Downloading a Configuration File over HTTPS

- The original startup configuration is overwritten with the new configuration file.
- A reboot is required for the new configuration to take effect.
- A syslog startup notification is generated automatically when the startup configuration is updated.
- The HTTPS client uses the device certificate.
- The remote end must serve the ICX startup, either by a web server or have a web service providing the configuration for the URL specified in the HTTP request.
- No special HTTP headers are required.
- Configuration download over HTTPS is not available in FIPS or CC mode.
- Only IPv4 HTTPS download is supported.

Downloading a Configuration File from a Web Server to the Startup Configuration

The following example copies an ICX configuration from the HTTPS server, specifying IP address 10.2.1.1 and the “cfg/backup.cfg” file name. Port number 876 is also specified.

```
device# copy https startup-config 10.2.1.1 cfg/backup.cfg port 876
```

The following example copies an ICX configuration from the HTTPS server, specifying IP address 10.1.1.1 and the “cfg/backup.cfg” file name. Because no port is specified, the default of 443 is used.

```
device# copy https startup-config 10.1.1.1 cfg/backup.cfg
```

Configuration Upload over HTTPS

Support has been added for uploading of a copy of the running configuration file or the startup configuration file from a FastIron device to an HTTPS server. HTTPS configuration upload works in a simple process.

After the **copy startup-config https** or **copy running-config https** command is executed specifying the URL to be used to upload ether the startup configuration or the running configuration to a remote HTTPS server, either the running configuration is generated or the startup configuration is read from flash memory. An HTTP PUT request is issued to the URL of the server, along with the contents of the running or startup configuration in the body of the request. The URL is in the form of path/filename. The remote server handles this request. The application on the server receives the configuration and stores it, typically in a file or a database. The server responds with status code 204, indicating success.

Considerations for Uploading a Configuration File over HTTPS

- The remote end must have a web service to upload the data in the HTTPS request and store it remotely in a database or file system. The web service must accept the URL specified in the HTTPS request.
- The HTTPS client uses the device certificate.
- The Content-Type header is set to plain text.
- Configuration upload over HTTPS is not available in FIPS or CC mode.

Uploading the Running Configuration to a Web Server

The following example uploads a copy of the running configuration file from a device to the HTTPS server, and specifies port 200.

```
device# copy running-config https 10.1.1.1 upload/backup.cfg port 200
```

Uploading the Startup Configuration to a Web Server

The following example uploads a copy of the startup configuration file from a device to the HTTPS server. Because no port is specified, the default of 443 is used.

```
device# copy startup-config https 10.1.1.1 backup/icx.cfg
```

Protecting against Denial of Service Attacks

- Denial of Service Protection Overview 253
- Protecting against Smurf Attacks 253
- Protecting against TCP SYN Attacks 255
- Defense against TCP Denial of Service Attacks on ICX 8200 Devices 258
- Defense against ICMP Denial of Service Attacks on ICX 8200 Switches 259
- Displaying Statistics from a DoS Attack 260
- Clear DoS Attack Statistics 261

Denial of Service Protection Overview

In a Denial-of-Service (DoS) attack, a router is flooded with useless packets for the purpose of slowing down or stopping normal operation.

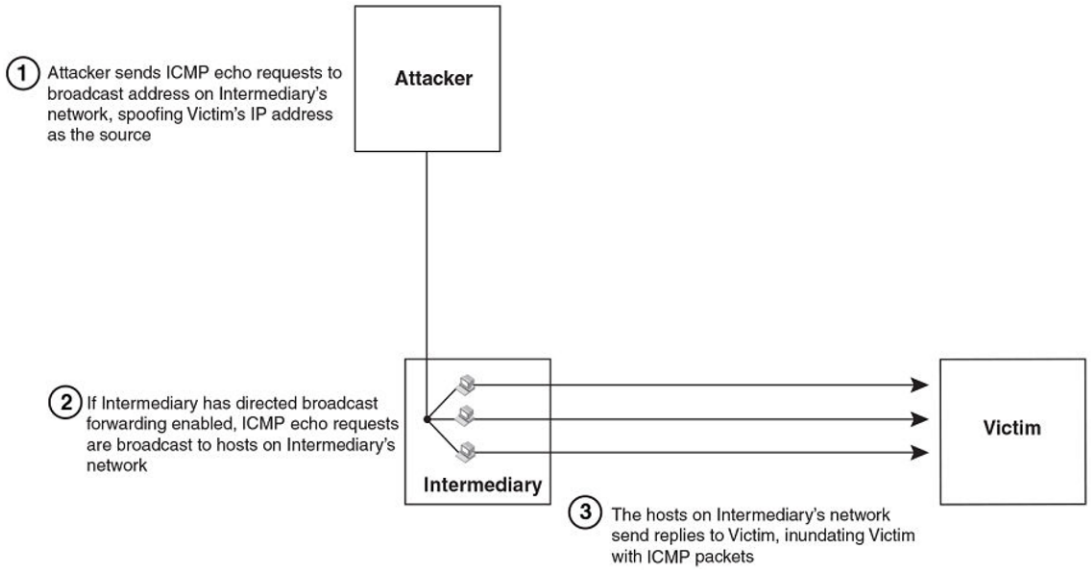
RUCKUS ICX devices include measures to defend against two types of DoS attacks: Smurf attacks and TCP SYN attacks.

- Smurf attacks—The 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—The attacker floods a host with TCP SYN packets that have random source IP addresses that fill up the connection queue, potentially causing service to 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 27 illustrates how a smurf attack works.

FIGURE 27 How a Smurf Attack Floods a Victim with ICMP Replies



The attacker sends an ICMP echo request packet to the broadcast address of an intermediary network. The ICMP echo request packet contains the spoofed address of a victim network as its source. When the ICMP echo request reaches the intermediary network, it is converted to a Layer 2 broadcast and sent to the hosts on the intermediary network. The hosts on the intermediary network then send ICMP replies to the victim network.

For each ICMP echo request packet sent by the attacker, a number of ICMP replies equal to the number of hosts on the intermediary network are sent to the victim. If the attacker generates a large volume of ICMP echo request packets, and the intermediary network contains a large number of hosts, the victim can be overwhelmed with ICMP replies.

Avoiding Being an Intermediary in a Smurf Attack

A smurf attack relies on the intermediary to broadcast ICMP echo request packets to hosts on a target subnet. When the ICMP echo request packet arrives at the target subnet, it is converted to a Layer 2 broadcast and sent to the connected hosts. This conversion takes place only when directed broadcast forwarding is enabled on the device.

To avoid being an intermediary in a smurf attack, make sure forwarding of directed broadcasts is disabled on the device. Directed broadcast forwarding is disabled by default. To disable directed broadcast forwarding, enter this command.

```
device(config)# no ip directed-broadcast
```

Avoiding Being a Victim in a Smurf Attack

You can configure the RUCKUS device to drop ICMP packets when excessive numbers are encountered as is the case when the device is the victim of a Smurf attack. You can set threshold values for ICMP packets that are targeted at the router itself or that pass through an interface and drop them when the thresholds are exceeded.

NOTE
ICX 8200 devices do not support the **ip icmp attack-rate burst-normal burst-max** command. Refer to Defense against ICMP Denial of Service Attacks on ICX 8200 Switches on page 259 for information on available ICMP DDOS options on ICX 8200 devices.

The number of incoming ICMP packets is measured and compared to the threshold values as follows:

- If the number of ICMP packets exceeds the **burst-normal** value, the excess ICMP packets are dropped.
- If the number of ICMP packets exceeds the **burst-max** value, all ICMP packets are dropped for the number of seconds specified by the **lockup** value. When the lockup period expires, the packet counter is reset, and measurement is restarted.

  NOTE
  The **burst-normal** value parameter can be from 20 through 10,000,000 Kbps. The **burst-max** value parameter can be from 20 through 10,000,000 Kbps. The **lockup** parameter can be from 1 through 10,000 seconds. This command is supported on Ethernet and Layer 3 interfaces.

Configuring Threshold Values for ICMP Packets Globally

Setting the threshold values for ICMP packets in global configuration mode.

1. Enter the global configuration mode to set the values globally.

   ```
   device# configure terminal
   ```

2. Set the threshold values for ICMP packets at the router.

   ```
   device(config)# ip icmp attack-rate burst-normal 2000 burst-max 2500 lockup 300
   ```

   In the above example, if incoming ICMP traffic exceeds 2000 Kbps, the excess packets are dropped. If the incoming ICMP traffic exceeds 2500 Kbps, the device drops all ICMP packets for the next 300 seconds.

Configuring ICMP Threshold Values on an Interface

Follow these steps to configure ICMP threshold values.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Specify the interface to be configured to enter interface configuration sub-mode.

```
device(config)# interface ethernet 1/3/11
```

3. Specify in Kbps the **burst-normal** and **burst-max** threshold values for ICMP packets. On the same line, specify the lockup period in seconds.

```
device(config-if-e1000-1/3/11)# ip icmp attack-rate burst-normal 2000 burst-max 2500 lockup 300
```

If the interface is part of a VLAN that has a router VE, you must configure ICMP attack protection at the VLAN level. Otherwise, you can configure this feature at the interface level as shown previously. When ICMP attack protection is configured at the VLAN level, it applies to routed traffic only. It does not affect switched traffic.

You must configure VLAN information for the port before configuring ICMP attack protection. You cannot change the VLAN configuration for a port on which ICMP attack protection is enabled.

To set threshold values for ICMP packets received on VLAN 2, enter the following commands.

```
device# configure terminal
device(config)# vlan 2
device(config-vlan-2)# ip icmp attack-rate burst-normal 5000 burst-max 10000 lockup 300
```

Protecting against TCP SYN Attacks

TCP SYN attacks exploit the process of how TCP connections are established to disrupt normal traffic flow. When a TCP connection starts, the connecting host first sends a TCP SYN packet to the destination host. The destination host responds with a SYN ACK packet, and the connecting host sends back an ACK packet. This process, known as a "TCP three-way handshake," establishes the TCP connection.

While waiting for the connecting host to send an ACK packet, the destination host keeps track of the as-yet incomplete TCP connection in a connection queue. When the ACK packet is received, information about the connection is removed from the connection queue. Usually there is not much time between the destination host sending a SYN ACK packet and the source host sending an ACK packet, so the connection queue clears quickly.

In a TCP SYN attack, an attacker floods a host with TCP SYN packets that have random source IP addresses. For each of these TCP SYN packets, the destination host responds with a SYN ACK packet and adds information to the connection queue. However, because the source host does not exist, no ACK packet is sent back to the destination host, and an entry remains in the connection queue until it ages out (after approximately a minute). If the attacker sends enough TCP SYN packets, the connection queue can fill up, and service can be denied to legitimate TCP connections.

To protect against TCP SYN attacks, you can configure the RUCKUS device to drop TCP SYN packets when excessive numbers are encountered. You can set threshold values for TCP SYN packets that are targeted at the router itself or that pass through an interface, and drop them when the thresholds are exceeded.

> **NOTE**
> ICX 8200 devices do not support the **ip tcp attack-rate burst-normal burst-max** command. Refer to Defense against TCP Denial of Service Attacks on ICX 8200 Devices on page 258 for information on available TCP DDOS options on ICX 8200 devices.

Incoming TCP SYN traffic is measured and compared to the threshold values as follows:

- If incoming TCP SYN packets exceed the **burst-normal** value, the excess TCP SYN packets are dropped.

- If incoming TCP SYN packets exceed 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 burst counter is reset, and measurement is restarted.

Configuring Threshold Values for TCP SYN Packets Globally

Setting threshold values for TCP SYN packets will protect against TCP SYN attack.

NOTE
The burst-normal value can be from 30 through 16,000,000 packets per second. The burst-max value can be from 30 through 16,000,000 packets per second. The lockup seconds can be from 1 through 10,000 seconds.

1. Enter the global configuration mode to set the values globally.

```
device# configure terminal
```

2. Set the burst-normal and burst-max threshold values for incoming TCP SYN traffic at the router. On the same line, configure the lockup time.

```
device(config)# ip tcp burst-normal 30 burst-max 100 lockup 300
```

In the example, if incoming TCP SYN traffic exceeds 30 packets per second, excess packets are dropped. If incoming TCP SYN traffic exceeds 100 packets per second, the device drops all TCP SYN packets for the next 300 seconds.

```
device(config)# configure terminal
device(config)# ip tcp burst-normal 30 burst-max 100 lockup 300
```

Configuring TCP SYN Threshold Values on an Interface

TCP SYN threshold values can be configured globally, at the physical interface level, or at the VLAN level. Perform the following steps to set threshold values at the interface level.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Specify the interface to be configured.

```
device(config)# interface ethernet 1/3/11
```

3. In interface configuration mode, specify the threshold values for TCP SYN traffic received in packets per second. On the same line, specify the lockup time in seconds.

```
device(config-if-e1000-1/3/11)# ip tcp burst-normal 30 burst-max 100 lockup 300
```

The following example configures the TCP/SYN attack protection at the VLAN level. If the interface is part of a VLAN that has a router VE, you must configure TCP/SYN attack protection at the VLAN level. When TCP/SYN attack protection is configured at the VLAN level, it applies to routed traffic only. It does not affect switched traffic.

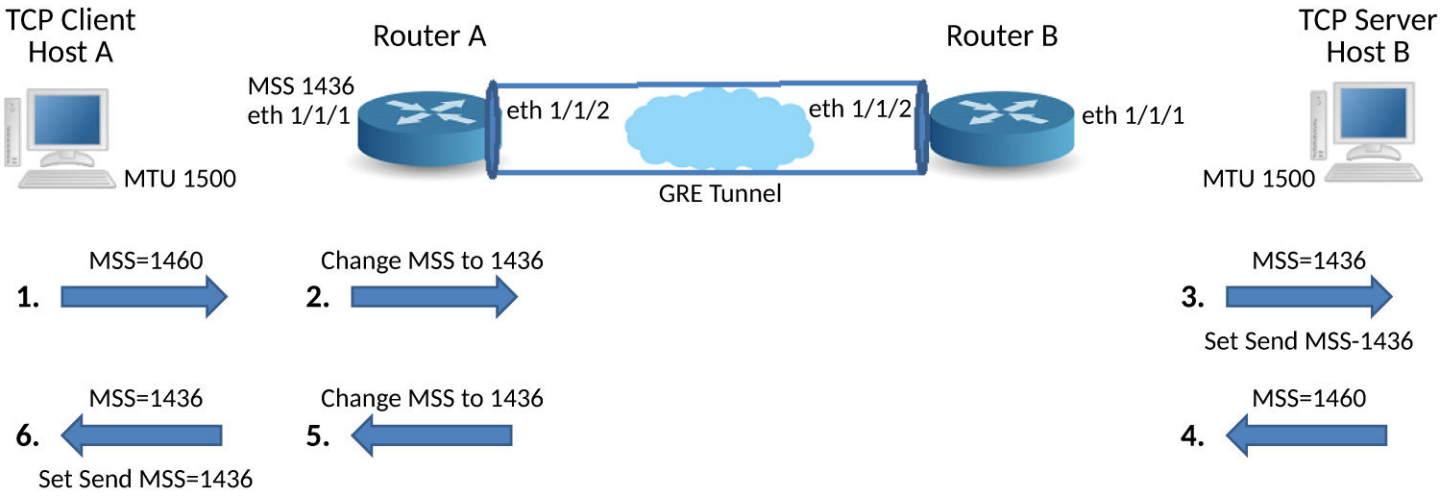
NOTE
You must configure VLAN information for the port before configuring TCP/SYN attack protection. You cannot change the VLAN configuration for a port on which TCP/SYN attack protection is enabled.

```
device# configure terminal
device(config)# vlan 2
device(config-vlan-2)# ip tcp burst-normal 5000 burst-max 10000 lockup 300
```

TCP MSS Adjustment Overview

TCP MSS adjustment modifies the maximum segment size (MSS) field from the TCP header in the TCP SYN/SYN-ACK packet during a three-way handshake. TCP MSS adjustment applies only to TCP connections that pass through the router. It does not apply to TCP connections originating from or destined to the router.

Example



Host A is a TCP client, Host B is the TCP server, and there are two routers, A and B. Router A has TCP MSS adjustment configured on ingress port Ethernet 1/1/1 with a specific value because there is a GRE tunnel in the path, which will add an additional 24 bytes of encapsulation. Router A modifies the TCP MSS adjustment value in the TCP SYN packet and forwards the packet. Host B receives the TCP SYN packet with the TCP MSS adjustment value set to the same value. Host B compares its MSS value to the received MSS value and sends the lower value to Host A as the "Send MSS" value. Host B send its own MSS value in the TCP SYN-ACK. In Router A, egress port Ethernet 1/1/1 is configured with the TCP MSS adjustment value. Router A modifies the MSS value in the TCP SYN-ACK and forwards the packet. Host A receives the TCP SYN-ACK and sends the "Send MSS" value to Host B.

Because Host A and Host B set a specific "Send MSS" value, neither of them will send a TCP data segment greater than a specific number of bytes to the other host. IP fragmentation of the TCP data packet is unnecessary eventhough GRE tunnel encapsulation occurs at Router A and Router B.

The same outcome can be achieved by configuring TCP MSS adjustment in the GRE tunnel interface in Router A. In this case, the MSS will be modified only for the packets traversing the tunnel. The remaining traffic will not be modified.

Impact on Existing Functionality

The TCP SYN attack feature is affected because of TCP MSS adjustment. Both features look for the same fields in the packet, so either one of the ACL rules can be hit.

Because TCP MSS adjustment has built-in rate limiting applied, it can be viewed as a superset. Higher priority is given to TCP MSS adjustment when both features are configured on a Layer 3 interface.

TCP MSS Adjustment Limitations

- When TCP MSS adjustment is enabled on an interface, some delay occurs in the TCP connection. Because the ICX device supports MSS adjustment on ingress and egress traffic of an interface as well as for both SYN and SYN-ACK packets, two packets are trapped to the CPU (Ingress SYN and egress SYN-ACK) for one TCP connection.
- TCP MSS adjustment is not supported for OpenFlow packets.
- When TCP SYN attack and TCP MSS adjustment are configured on an interface, TCP MSS adjustment takes the higher priority.
- When policy-based routing (PBR) and TCP MSS adjustment are both applied, the TCP connections that go through PBR may be delayed, even if the outgoing interface of the PBR route does not have the TCP MSS adjustment configuration.
- When MSS is configured on the outgoing interface that PBR points to, but not on the outgoing interface that Layer 3 lookup points to, the packet is not trapped to the CPU. The MSS is not modified for the packet. For example, suppose the normal route lookup for an incoming packet with the destination address 10.10.10.10 goes through interface 1/1/10, but PBR is configured for redirecting a packet with destination address 10.10.10.10 to interface 1/1/1. If TCP MSS is configured on interface 1/1/1, MSS is not modified for the TCP SYN packet arriving with destination address 10.10.10.10 even though it traverses MSS-configured interface 1/1/1.
- TCP MSS adjustment ACL rules are given higher priority than the user-defined ACL rules. Therefore, even if the user-defined rule is applied to drop TCP packets, a TCP packet comes to the CPU because of the TCP MSS adjustment rule and is dropped in software.
- For ECMP destination routes, there is no guarantee that the packet will go through the path chosen in hardware. The outgoing TCP SYN/SYN-ACK packet will be trapped to the CPU, and the MSS is modified according to the software route lookup.
- When TCP MSS adjustment is applied on a physical, VE, or LAG interface, the MSS is modified only for plain IP or IPv6 traffic. The MSS for tunneled or encrypted packets is not modified.
- The MSS value cannot be modified for an AH-only IPsec packet (unencrypted). The MD5/SHA-1 hash cannot be recomputed because the security keys to compute the hash value are unknown.
- TCP MSS adjustment configuration is not supported on loopback or management interfaces.
- TCP MSS adjustment is supported only on the Layer 3 interface.

Defense against TCP Denial of Service Attacks on ICX 8200 Devices

Commands are available on ICX 8200 switches to rate limit TCP control traffic or to drop specific kinds of irregular TCP SYN control packets.

Rate Limiting TCP Control Traffic

Use the **ip tcp burst-max** command as shown to limit TCP control traffic flow. Valid values are 125 through 100000 packets per second.

The following example limits the rate of TCP SYN control packets to 1000 packets per second.

```
ICX8200# configure terminal
ICX8200(config)# ip tcp burst-max 1000
```

The command can also be used on a specific interface. The following example limits TCP SYN control packets to 125 packets per second on port 1/2/1.

```
ICX8200# configure terminal
ICX8200(config)# interface ethernet 1/2/1
ICX8200(config-if-e1000-1/2/1)# ip tcp burst-max 125
```

Dropping Irregular TCP Control Packets

On ICX 8200 switches, the following commands can be entered in global configuration mode as shown to drop suspect, malformed TCP control packets that may indicate a DOS attack. The commands can be configured on the same device as needed. To drop all malformed TCP control packets in this set, use the **ip tcp tcp-all** command.

The following example drops IPv4 and IPv6 TCP control packets sent using the MAC multicast mechanism.

```
ICX8200(config)# ip tcp over-mac-multicast
```

The following example drops TCP control packets with all flags set to zero.

```
ICX8200(config)# ip tcp zero-flags
```

The following example drops TCP control packets with both the SYN and URG-PSH flags set.

```
ICX8200(config)# ip tcp fin-urg-psh
```

The following example drops TCP control packets with both the SYN and FIN flags set.

```
ICX8200(config)# ip tcp syn-fin
```

The following example drops TCP control packets with both the SYN and RST flags set.

```
ICX8200(config)# ip tcp syn-rst
```

The following example drops TCP control packets with the source or destination port set to zero.

```
ICX8200(config)# ip tcp port-zero
```

The following example drops all malformed TCP control packets in this set.

```
ICX8200(config)# ip tcp tcp-all
```

Defense against ICMP Denial of Service Attacks on ICX 8200 Switches

To prevent DOS attacks on ICX 8200 devices, you can rate limit ICMP pings to the CPU. You can also drop irregular ICMP packet fragments.

Rate Limiting ICMP Control Traffic

Use the **ip icmp burst-max** command as shown to limit ICMP pings to the CPU. Valid values are 20 through 10000000 kbps.

The following example limits the rate of ICMP pings to the CPU to 50 Kbps.

```
ICX8200# configure terminal
ICX8200(config)# ip icmp burst-max 50
```

The command can also be used on a specific interface. The following example limits ICMP pings to the CPU to 20 Kbps on port 1/2/3.

```
ICX8200# configure terminal
ICX8200(config)# interface ethernet 1/2/3
ICX8200(config-if-e1000-1/2/3)# ip icmp burst-max 20
```

Dropping Suspicious ICMP Packet Fragments

Use the following command to drop irregular, suspect ICMP packet fragments in order to prevent DOS attacks.

```
ICX8200# configure terminal
ICX8200(config)# ip icmp icmp-fragments
```

Displaying Statistics from a DoS Attack

To display information about the dropped ICMP and TCP SYN packets due to increased burst thresholds, enter the **show statistics dos-attack** command. The following example displays the output for most ICX devices.

```
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
```

DOS Show Commands for ICX 8200 Devices

To display information about TCP SYN packets that exceeded burst thresholds on an ICX 8200 device, enter the **show statistics dos-attack tcp** command as shown in the following example.

```
ICX8200# show statistics dos-attack tcp
Global Attack Statistics
------------------------
TCP-SYN
-----------------------
Flags drop count Drop_count
------------------ --------------
52586 61240
Interface Attack Statistics
---------------------------
TCP-SYN
--------------------------------
Port Dropped pkts
--------- ----------------------
1/1/10 61240
```

To display information about ICMP packets that exceeded burst thresholds on an ICX 8200 device, enter the **show statistics dos-attack icmp** command as shown in the following example.

```
ICX8200# show statistics dos-attack icmp
Global Attack Statistics
------------------------
ICMP
----------------------
Dropped pkts
------------
12691072
Interface Attack Statistics
```

```
---------------------------
ICMP
--------
Port Dropped pkts
--------- ---------------
1/1/10 1191808
```

To display information about ICMP fragments received on an ICX 8200 device, enter the **show statistics dos-attack icmp fragments** command as shown in the following example.

```
ICX8200# show statistics dos-attack icmp fragments
icmp fragment Statistics
------------------------
Drop_count
------------------
32396
```

Clear DoS Attack Statistics

To clear statistics about ICMP and TCP SYN packets, enter the **clear statistics dos-attack** command.

```
device# clear statistics dos-attack
```

IPv6 RA Guard

- Securing IPv6 Address Configuration....263
- IPv6 RA Guard Overview....263
- Configuration Notes and Feature Limitations for IPv6 RA Guard....264
- Configuring IPv6 RA Guard....264
- Example of Configuring IPv6 RA Guard....265

Securing IPv6 Address Configuration

In an 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-the-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 an IPv6 RA guard policy and associate criteria such as whitelist, prefix list, and maximum preference 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

A 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.

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

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 trusted, untrusted, or host port configuration to take effect, 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

- 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 to associate a prefix list to 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. A RA guard policy name can be a maximum of 512 characters. 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) Associate an already defined prefix list with the RA guard policy using the **prefix-list** command in the RA guard policy configuration mode. You must provide the name of an IPv6 prefix list already configured using the **ipv6 prefix-list** command. Associate a prefix-list with an RA guard policy using the **prefix-list** command.
6. (Optional) Set the preference for RA packets using the **preference-maximum** command in the RA guard policy configuration mode.
7. Apply the RA guard policy to a VLAN using the **ipv6 raguard vlan** command in the global configuration mode. You can associate only one RA guard policy with a VLAN.
8. (Optional) Enable logging using the **logging** command in the RA guard policy configuration mode. If logging is enabled, you can verify the logs like RAs dropped, permitted, count for dropped packets, and reasons for the drop. Logging increases the CPU load and, for higher traffic rates, RA packets drop due to congestion if they are received at the line rate.
9. (Optional) Verify the RA guard configuration using the **show ipv6 raguard** command.
10. (Optional) Clear the RA packet counter using the **clear ipv6 raguard** command.
11. (Optional) Verify the RA packet counts using the **show ipv6 raguard counts command. Logging has to be enabled to verify the counts.**

Example of Configuring IPv6 RA Guard

The following sections describe how to configure IPv6 RA guard on a device or in a network.

Example: Configuring IPv6 RA Guard on a Device

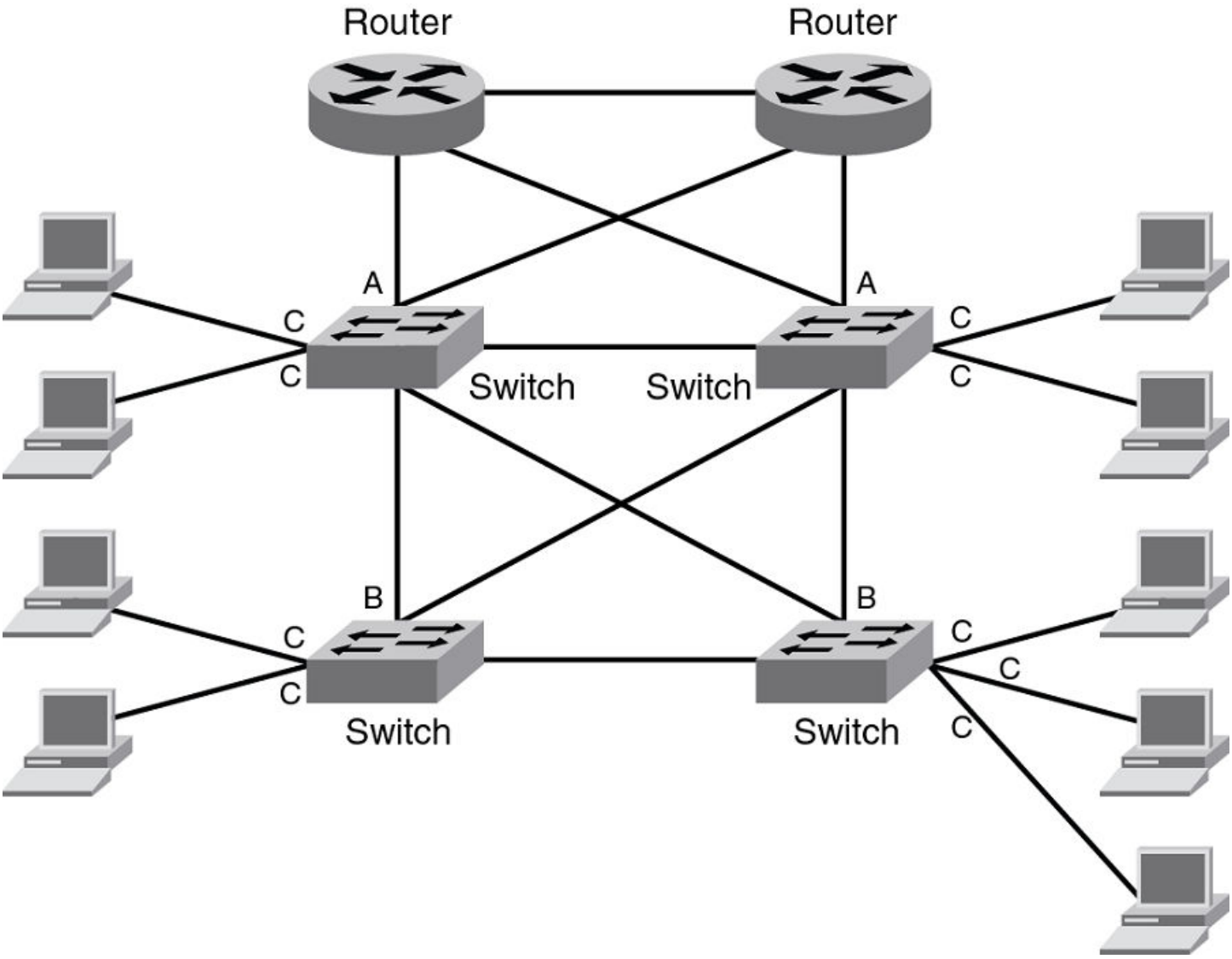
The following example shows how to configure RA guard on a device.

```
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:1
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:3
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:10
device(config)# ipv6 raguard policy policy1
device(ipv6-RAG-policy policy1)# whitelist 1
device(ipv6-RAG-policy policy1)# prefix-list raguard-prefix1
device(ipv6-RAG-policy policy1)# preference-maximum medium
device(ipv6-RAG-policy policy1)# logging
device(ipv6-RAG-policy policy1)# exit
device(config)# interface ethernet 1/1/1
device(config-int-e1000-1/1/1)# raguard untrust
device(config-int-e1000-1/1/1)# exit
device(config)# ipv6 raguard vlan 1 policy policy1
```

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 28 IPv6 RA Guard Configuration in a Network



Configuring Port A:

Configure port A as a trusted port.

```
device(config)# interface ethernet 1/1/1
device(config-int-e1000-1/1/1)# raguard trust
```

Configuring Port C:

On port C, create an RA Guard policy with no other options and associate the policy with a VLAN of which C is a member of. This helps block all RAs from C ports.

```
device(config)# ipv6 raguard policy policyC
device(ipv6-RAG-policy policyC)# exit
device(config)# ipv6 raguard vlan 1 policy policyC
```

Configuring Port B:

On port B create an RA Guard policy with supported whitelist. This helps to permit RAs from only those sources. Associate a whitelist or prefix list with the RA guard policy.

```
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:10
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:5
device(config)# ipv6 raguard whitelist 1 permit fe80:db8::db8:12
device(config)# ipv6 prefix-list raguard-prefix-list1 permit 2001::db8::/16
device(config)# ipv6 raguard policy policyB
device(ipv6-RAG-policy policyB)# whitelist 1
device(ipv6-RAG-policy policyB)# prefix-list raguard-prefix-list1
device(ipv6-RAG-policy policyB)# exit
device(config)# interface ethernet 1/1/2
device(config-int-e1000-1/1/2)# raguard untrust
device(config-int-e1000-1/1/2)# exit
device(config)# ipv6 raguard vlan 2 policy policyB
```

Example: Verifying the RA Guard Configuration

To view the RA guard packet counts, use the **show ipv6 raguard counts** command.

```
device# show ipv6 raguard counts policyB
DROPPED-host port:0
DROPPED-whitelist:3
DROPPED-prefixlist:1
DROPPED-max pref:1
DROPPED-trusted port:2
DROPPED-untrusted port:1
```

To verify the RA guard policy configuration, enter the **show ipv6 raguard policy all** command.

```
device# show ipv6 raguard policy all
policy:policyC
        whitelist:0
        max_pref:medium
policy:policyB
        whitelist:1
```

To verify the configuration of RA guard whitelists, enter the **show ipv6 raguard whitelist all** command.

```
device#show ipv6 raguard whitelist all
whitelist #1 : 3 entries
        permit fe80:db8::db8:10/128
        permit fe80:db8::db8:5/128
        permit fe80:db8::db8:12/128
```

To verify the RA guard configuration for a specific VLAN, enter the **show ipv6 raguard vlan** command followed by the ID of VLAN you want to check.

The following example displays output for a VLAN.

```
device# show ipv6 raguard vlan 320
VLAN     Policy
-----    ------
320      policy650
```

To check IPv6 RA guard in the running configuration, enter the **show running-config** command. The following example displays all IPv6 RA Guard whitelists and policies configured on the ICX device and verifies that two VLANs (VLAN 100 and VLAN 200) have IPv6 RA Guard policies bound to them.

```
device(config)# show running-config
Current configuration:
!
ver 09.0.00
!
stack unit 1
```

```
  module 1 icx7650-48zp-port-management-module
  module 2 icx7600-qsfp-2port-80g-module
  module 3 icx7650-100g-2port-200g-module
  stack-port ethernet 1/3/1
  stack-port ethernet 1/3/2
stack mac 609c.9f52.29ae
!
hitless-failover enable
!
!
ipv6 raguard whitelist 10 permit 1122:3344::5566:3344/64
ipv6 raguard whitelist 10 permit 1133:3344::5566:3344/96
ipv6 raguard whitelist 10 permit 1144:3344::5566:3344
ipv6 raguard whitelist 10 permit 1222:3344::5566:3344/64
ipv6 raguard whitelist 10 permit 1233:3344::5566:3344/96
ipv6 raguard whitelist 10 permit 1244:3344::5566:3344
ipv6 raguard whitelist 10 permit 1322:3344::5566:3344/64
ipv6 raguard whitelist 10 permit 1333:3344::5566:3344/96
!
ipv6 raguard whitelist 20 permit 1122:3344::5566:3344/64
ipv6 raguard whitelist 20 permit 1133:3344::5566:3344/96
ipv6 raguard whitelist 20 permit 1144:3344::5566:3344
ipv6 raguard whitelist 20 permit 1222:3344::5566:3344/64
ipv6 raguard whitelist 20 permit 1233:3344::5566:3344/96
!
ipv6 raguard whitelist 22 permit 1122:3344::5566:3344/64
ipv6 raguard whitelist 22 permit 1133:3344::5566:3344/96
ipv6 raguard whitelist 22 permit 1144:3344::5566:3344
ipv6 raguard whitelist 22 permit 1222:3344::5566:3344/64
ipv6 raguard whitelist 22 permit 1233:3344::5566:3344/96
ipv6 raguard whitelist 22 permit 1244:3344::5566:3344
!
ipv6 raguard whitelist 31 permit 1122:3344::5566:3344/64
ipv6 raguard whitelist 31 permit 1133:3344::5566:3344/96
ipv6 raguard whitelist 31 permit 1144:3344::5566:3344
ipv6 raguard whitelist 31 permit 1222:3344::5566:3344/64
ipv6 raguard whitelist 31 permit 1233:3344::5566:3344/96
ipv6 raguard whitelist 31 permit 1244:3344::5566:3344
ipv6 raguard whitelist 31 permit 1322:3344::5566:3344/64
!
ipv6 raguard whitelist 42 permit 1122:3344::5566:3344/64
ipv6 raguard whitelist 42 permit 1133:3344::5566:3344/96
ipv6 raguard whitelist 42 permit 1144:3344::5566:3344
ipv6 raguard whitelist 42 permit 1222:3344::5566:3344/64
ipv6 raguard whitelist 42 permit 1233:3344::5566:3344/96
ipv6 raguard whitelist 42 permit 1244:3344::5566:3344
ipv6 raguard whitelist 42 permit 1322:3344::5566:3344/64
ipv6 raguard whitelist 42 permit 1333:3344::5566:3344/96
!
ipv6 raguard whitelist 48 permit 1122:3344::5566:3344/64
ipv6 raguard whitelist 48 permit 1133:3344::5566:3344/96
ipv6 raguard whitelist 48 permit 1144:3344::5566:3344
ipv6 raguard whitelist 48 permit 1222:3344::5566:3344/64
ipv6 raguard whitelist 48 permit 1233:3344::5566:3344/96
ipv6 raguard whitelist 48 permit 1244:3344::5566:3344
ipv6 raguard whitelist 48 permit 1322:3344::5566:3344/64
ipv6 raguard whitelist 48 permit 1333:3344::5566:3344/96
!
ipv6 raguard whitelist 63 permit 1122:3344::5566:3344/64
ipv6 raguard whitelist 63 permit 1133:3344::5566:3344/96
ipv6 raguard whitelist 63 permit 1144:3344::5566:3344
ipv6 raguard whitelist 63 permit 1222:3344::5566:3344/64
ipv6 raguard whitelist 63 permit 1233:3344::5566:3344/96
ipv6 raguard whitelist 63 permit 1244:3344::5566:3344
ipv6 raguard whitelist 63 permit 1322:3344::5566:3344/64
ipv6 raguard whitelist 63 permit 1333:3344::5566:3344/96
!
!
ipv6 raguard policy pol20
 whitelist 10
 prefix-list prefix1
 maximum-preference medium
```

```
 logging
!
ipv6 raguard policy pol28
 whitelist 22
 prefix-list prefix28
 maximum-preference medium
 logging
!
ipv6 raguard policy pol31
 whitelist 31
 prefix-list prefix2
 maximum-preference low
 logging
!
ipv6 raguard policy pol63
 whitelist 63
 logging
!
ipv6 raguard policy pol2
 whitelist 20
 prefix-list prefix20
 maximum-preference medium
!
ipv6 raguard policy pol40
 prefix-list prefix40
 maximum-preference low
 logging
!
ipv6 raguard policy pol37
 whitelist 42
 prefix-list prefix1
 maximum-preference medium
 logging
!
ipv6 raguard policy pol56
 whitelist 48
 prefix-list prefix1
 logging
!
ipv6 raguard policy pol18
 whitelist 20
 prefix-list prefix18
 maximum-preference low
 logging
!
ipv6 raguard policy pol58
 logging
!
ipv6 raguard policy pol39
 whitelist 22
 prefix-list prefix39
 maximum-preference medium
 logging
!
!
ipv6 raguard vlan 100 policy pol20
ipv6 raguard vlan 200 policy pol28
!
!
!
```

Joint Interoperability Test Command

- JITC Overview..... 271

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. For enhanced security, the JITC mode uses AES-CTR (Counter) encryption mode for SSH instead of AES-CBC mode.

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 Specifying which View is Notified of Traps" section in the *SNMP* chapter of the *RUCKUS FastIron Management Configuration Guide*.

OpenSSL License

- OpenSSL License..... 273

OpenSSL License

Copyright (c) 1998-2001 The OpenSSL Project. All rights reserved.

1. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
2. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
3. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation or other materials provided with the distribution.
4. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
5. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org .
6. Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.
7. Redistributions of any form whatsoever must retain the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)"

   THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT "AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Original SSLeay License

Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com) All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com). The implementation was written so as to conform with Netscapes SSL. This library is free for commercial and non-commercial use as long as the following conditions are aheared to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com). Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

1. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
2. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.

3. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes cryptographic software written by Eric Young(eay@cryptsoft.com)" The word 'cryptographic' can be left out if the rouines from the library being used are not cryptographic related.
4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgment: "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

   THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. The licence and distribution terms for any publically available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence.

Keychain Module

- Keychain Module Overview....275
- OSPF Keychain Authentication....276
- TCP Keychain Options....277
- MKA Keychain Overview and Considerations....280

Keychain Module Overview

Keychain is a utility module that can be used by any application or routing protocol that requires authentication keys to establish secure communication with peers and validate the control packets.

The keychain module provides a secure infrastructure to prevent unauthorized routing updates to the network and ensure that only trusted routers participate in routing updates. Apart from authentication, the keychain module provides a mechanism to ensure key rollover based on the duration specified for each key.

How the Keychain Module Works

The keychain module is independent of the protocols or applications that use it. The protocol packets use one of the active keys from the designated keychain profile. The keychain profile may have multiple keys with different attributes, such as authentication algorithms, passwords, and lifetimes.

Each key in the keychain has a lifetime associated with it. A key is considered active if it is within the configured time range and has an authentication algorithm and a password. When a key expires, the keychain module notifies the application.

NOTE
All participating routers must have Network Time Protocol (NTP) enabled before setting their lifetimes.

Keys cannot be used for authentication if they are not active. To allow for key rollover, the lifetime of the keys must be configured so that the key activation periods overlap and active keys are available at any time. As an option, a tolerance value can be configured for the accept-keys and send-keys in the keychain to extend the lifetime of the keys. The tolerance period smooths the transition to a new key.

An application uses a specified key and algorithm from the keychain module to generate a message digest for sent messages. It uses the message digest to validate packets when they are received.

When an application requests keys from the keychain module for sending and accepting the packets, the keychain module supplies all the active keys from the keychain, and the application picks the desired key based on criteria specific to the protocol or application. The sending peer picks the key based on its lifetime and a cryptographic algorithm that matches its criteria. The receiving peer selects the key it uses for authentication based on the incoming key ID. When a keychain is configured for a protocol, all the packets generated by the protocol, such as routing updates and hello packets, are validated with that key ID. Because the same key ID is used to validate packets received, it is imperative that the neighbors and participating routers have the same configuration at the other end.

Components of a Keychain

NOTE
For different protocols, keychain options may vary as described under the specific protocol.

A keychain consists of the following components:

- Keychain profile: Each keychain is identified by a user-configured profile name. A maximum of 64 keychains can be configured.
- Keys: Keys are added to the keychain profile by specifying key IDs. Each key ID within a keychain has its own properties, such as a password, authentication algorithm, send lifetime, and accept lifetime. A key is considered valid only if the key lifetime has not expired, and the password and authentication algorithm are specified. A maximum of 1024 keys can be configured across all the keychains.

  For each protocol, the key ID must be within a valid range. For example, the valid range of key IDs for OSPFv2 is 1 through 255. The application that uses the keychain module can reject the key IDs that are outside the permitted range. However, the keychain module does not place any restrictions on key ID configuration.
- Authentication algorithm: Each key must have an authentication algorithm. The application or protocol chooses the cryptographic algorithm that matches its criteria. The following algorithms are supported:
  - AES-128-CMAC
  - HMAC-SHA-1
  - HMAC-SHA-256
  - MD5
  - SHA-1
  - SHA-256
- Password: Each key must have a password in encrypted form for the cryptographic algorithm.
- Lifetime of key: Each key in the keychain has send and accept lifetimes. A key is considered active if it is within a configured time range. The lifetime of the key also depends on the tolerance value.
  - Send lifetime: The time period during which the key on a keychain becomes active and is valid to be sent.
  - Accept lifetime: The time period during which the key on a keychain becomes active and is received as valid.
- Tolerance: The tolerance value extends the lifetime of keys beyond the configured active lifetime. A key is considered valid even when it is in the tolerance period. If the tolerance value is configured, the start time of the key is advanced (start time minus tolerance), and the end time is moved further ahead (end time plus tolerance), unless the end time is set to be infinite.

OSPF Keychain Authentication

Applications such as OSPF can benefit from a keychain module that provides hitless authentication key rollover, which overcomes the limitation of a static configuration in authentication methods that require manual intervention to change the key periodically.

For each OSPF protocol packet, a key is used to generate and verify a message digest. The key is valid for the duration of the protocol with no option to change the key string or authentication algorithm automatically. The keychain module, which functions as a container for keys with different attributes, allows OSPF to choose the key that best suits its criteria and automatically change the key ID, password, and cryptographic algorithm without manual intervention. OSPFv2 and OSPFv3 authentication using the keychain is configured with the **ip ospf authentication keychain** or **ipv6 ospf authentication keychain** commands. For more information on configuring OSPFv2 and OSPFv3 authentication to use the keychain module, refer to the *RUCKUS FastIron Layer 3 Routing Configuration Guide*.

Configuring a Keychain Module

Complete the following steps to configure a keychain module.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure a keychain profile.

```
device(config)# keychain ruckus
```

3. (Optional) Configure the tolerance value for the keys.

   The following example configures a 10,000 second tolerance period for the keychain profile "ruckus."

```
device(config-ruckus)# tolerance 10000
```

4. Configure a key by specifying a key identifier.

```
device(config-ruckus)# key-id 1
```

5. Configure the authentication algorithm to be used for the key.

   Authentication algorithms HMAC-SHA-1, HMAC-SHA-256, MD5, SHA-1, and SHA-256 are supported. The application or protocol chooses the cryptographic algorithm that matches its criteria.

   The following example configures MD5 as the authentication algorithm for key 1.

```
device(config-ruckus-key-1)# authentication-algorithm md5
```

6. Configure the password to be used for the key.

```
device(config-ruckus-key-1)# password abc
```

7. Configure the time period during which the key on a keychain is active and can be received as a valid key.

   The end time can be configured as one of the following options: duration in seconds, infinite, or date and time format (mm-dd-yy hh:mm:ss). The maximum lifetime in seconds is 2147483648.

   The following example configures the accept lifetime of key 1 to start on November 10, 2019 at 10:10 am and 10 seconds and to end 10,000 seconds later.

```
device(config-ruckus-key-1)# accept-lifetime start 11-10-19 10:10:10 end 10000
```

8. Configure the time period during which the key on a keychain becomes active and is valid to be sent.

   The end time can be configured as one of the following options: duration in seconds, infinite, or date and time format (mm-dd-yy hh:mm:ss).

   The following example configures key 1 to be active and available for sending from November 10, 2019 at 10:10 am and 10 seconds, with no expiration.

```
device(config-ruckus-key-1)# send-lifetime start 11-10-19 10:10:10 end infinite
```

TCP Keychain Options

The following commands are available only for TCP keychains:

- **authentication-algorithm aes-128-cmac**: Configures the AES-128-CMAC algorithm for TCP authentication. The **authentication-algorithm hmac-sha-1** command is also supported. The **no** form of the command removes the configuration.

- **accept-ao-mismatch**: Determines whether the ICX device accepts or denies TCP segments received with a mismatch in TCP-AO support between the TCP peers. By default, mismatched segments are accepted. When you use the **no** form of the command to disable the option, TCP-AO packets with a mismatch are discarded. For example, if the authentication option (AO) is not configured for a particular TCP BGP/MSDP peer connection and the ICX device receives over that connection a segment from a peer with the authentication option set, the ICX device determines how the mismatch is handled based on the command option. If the ICX device has the **no accept-ao-mismatch** option configured, no session will be established with the peer.
- **include-tcp-options**: Determines whether all TCP options are included in the Message Authentication Code (MAC) calculation. By default, all TCP options are included. When the **no** form of the command is configured, only the TCP-AO option is included in the MAC calculation.
- **send-id**: Configures the identifier sent in an outgoing TCP segment. Valid decimal values are 0 through 255. The **no** form of the command removes the identifier.
- **receive-id**: Configures the identifier to be compared with the key identifier received in a TCP segment. Valid decimal values are 0 through 255. The **no** form of the command removes the identifier.

TCP Keychain Considerations

It is not advisable to change the TCP keychain configuration for active connections.

The TCP keychain can be used in conjunction with routing protocols such as BGP and MSDP.

BGP and MSDP peer sessions must be cleared for the new TCP-AO configuration to take effect. Refer to the *RUCKUS FastIron Layer 3 Routing Configuration Guide* to configure BGP to use TCP-AO settings. Refer to the *RUCKUS FastIron IP Multicast Configuration Guide* to configure MSDP to use TCP-AO settings.

TCP Keepalive is enabled for all TCP-AO-enabled connections and cannot be disabled, even if disabled at the global level.

TCP-AO and TCP MD5 cannot be used in the same connection.

In an ICX stack, TCP authentication options are applicable only for the TCP connections created from the active-controller.

Configuring TCP Keychain Options

1. Enter global configuration mode.

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

2. Create a TCP keychain.

```
device(config)# keychain mykeychain tcp
device(config-keychain-tcp-mykeychain)#
```

   The command places the ICX device in TCP keychain configuration mode, where TCP authentication options are available.

3. Configure a key by specifying a key identifier.

```
device(config-keychain-tcp-mykeychain)# key-id 1
```

4. Specify the authentication algorithm to be used.

```
device(config-keychain-tcp-mykeychain-key-1)# authentication-algorithm aes-128-cmac
```

NOTE
The **hmac-sha-1** algorithm is also available for TCP authentication.

NOTE
Use the **no authentication-algorithm aes-128-cmac** command to remove the algorithm.

5. Specify whether TCP packet segments received with mimatched AO settings will be accepted or discarded.

```
device(config-keychain-tcp-mykeychain-key-1)# no accept-ao-mismatch
```

The example configures the ICX device to discard TCP segments with mismatched AO settings.

NOTE
By default, mismatched TCP segments are accepted.

6. Specify whether the MAC calculation includes all TCP options.

```
device(config-keychain-tcp-mykeychain-key-1)# include-tcp-options
```

NOTE
By default, all TCP options are included in the MAC calculation. To include only the TCP AO options in the calculation, use the **no include-tcp-options** command.

7. Configure the ID to be used in transmitted TCP packets. Enter a decimal value from 0 through 255.

```
device(config-keychain-tcp-mykeychain-key-1)# send-id 1
```

NOTE
This value must match the receive-id configured at the other end of the TCP connection.

8. Configure the ID to be compared to the key identifier in TCP packet segments received by the ICX device. Enter a decimal value from 0 through 255.

```
device(config-keychain-tcp-mykeychain-key-1)# recv-id 2
```

NOTE
This value must match the send-id configured at the other end of the TCP connection.

9. Configure the time period during which the key on a keychain is active and can be received as a valid key.

The end time can be configured as one of the following options: duration in seconds, infinite, or date and time format (mm-dd-yy hh:mm:ss). The maximum lifetime in seconds is 2147483648.

The following example configures the accept lifetime of key 1 to start on November 10, 2019 at 10:10 am and 10 seconds and to end 10,000 seconds later.

```
device(config-keychain-tcp-mykeychain-key-1)# accept-lifetime start 11-10-19 10:10:10 end 10000
```

10. Configure the time period during which the key on a keychain becomes active and is valid to be sent.

    The end time can be configured as one of the following options: duration in seconds, infinite, or date and time format (mm-dd-yy hh:mm:ss).

    The following example configures key 1 to be active and available for sending from November 10, 2019 at 10:10 am and 10 seconds, with no expiration.

    ```
    device(config-keychain-tcp-mykeychain-key-1)# send-lifetime start 11-10-19 10:10:10 end infinite
    ```

11. (Optional) Enable logging of TCP authentication option messages.

    ```
    device(config-keychain-tcp-mykeychain-key-1)# exit
    device(config-keychain-tcp-mykeychain)# exit
    device(config)# logging enable tcp-ao
    ```

12. (Optional) Verify the TCP keychain configuration.

    ```
    device(config)# show keychain name mykeychain
    Keychain: mykeychain
    TCP-AO: TRUE
    Key-id : 1
    Auth-Algorithm: aes-128-cmac
    Key-String : *******
    Send-id : 1
    Recv-id : 2
    include-tcp-options : YES
    accept-ao-mismatch : NO
    Send Lifetime:-
    Start : 11-10-19 10:10:10 End : Infinite
    Active : No TimeToActive: 27583321 sec
    Timezone : Local
    Accept Lifetime:-
    Start : 11-10-19 10:10:10  End : 10000
    Active : No TimeToActive: 27583321 sec
    Timezone : Local
    ```

    NOTE
    You can use the **show keychain tcp** or the **show keychain name** *name* command to verify the TCP keychain settings.

The following example configures a TCP keychain and underlying options.

```
device# configure terminal
device(config)# keychain mykeychain tcp
device(config-keychain-tcp-mykeychain)# key 1
device(config-keychain-tcp-mykeychain-key-1)# authentication-algorithm aes-128-cmac
device(config-keychain-tcp-mykeychain-key-1)# no accept-ao-mismatch
device(config-keychain-tcp-mykeychain-key-1)# include-tcp-options
device(config-keychain-tcp-mykeychain-key-1)# send-id 1
device(config-keychain-tcp-mykeychain-key-1)# recv-id 2
device(config-keychain-tcp-mykeychain-key-1)# accept-lifetime start 11-10-19 10:10:10 end 10000
device(config-keychain-tcp-mykeychain-key-1)# send-lifetime start 11-10-19 10:10:10 end infinite
device(cconfig-keychain-tcp-mykeychain-key-1)# exit
device(config-keychain-tcp-mykeychain)# exit
```

MKA Keychain Overview and Considerations

Keychains are an alternative to configuring a single pre-shared key on each MACsec interface. You can configure a maximum of eight MKA keychains, and each keychain can contain a maximum of 32 configured keys. Each key contains a pre-configured password, authentication algorithm, and a send lifetime configuration.

Keep the following considerations in mind when configuring MKA keychains.

- When you use the keychain feature for a MACsec interface, the authentication algorithm configured under **mka-keychain** is used. The authentication algorithm configured under **mka-cfg-group** is not used.
- An MKA keychain cannot be modified after it is bound to an interface.
- An MKA keychain does not support a combination of AES-128 and AES-256 algorithms. Only one of the two algorithms can be configured in the same keychain.
- The 'accept-lifetime' configuration under the keychain module configuration does not apply to the MACsec keychain.
- The interval between Start and End lifetime for a configured MKA key must be a minimum of 120 seconds.
- The interval between activation of two successive MKA keys must be a minimum of 120 seconds.
- All invalid keys in the MKA keychain will be ignored.

Creating and Configuring an MKA Keychain

Perform the following steps to configure an MKA keychain.

1. In global configuration mode, enter the **keychain** command followed by the keychain name and the keyword **mka**.

   ```
   device# configure terminal
   device(config)# keychain sample mka
   device(config-keychain-mka-sample)#
   ```

   The keychain is created, and the device is placed in MKA keychain configuration mode.

2. Configure the key identifier. Valid values are from 1 through 4294967296.

   ```
   device(config-keychain-mka-sample)# key-id 100
   ```

   The key is created, and the device is placed in MKA key configuration mode.

3. Configure the authentication algorithm for the key. Options are **authentication-algorithm aes-128-cmac** and **authentication-algorithm aes-256-cmac**.

   ```
   device(config-keychain-mka-sample-key-100)# authentication-algorithm aes-128-cmac
   ```

4. Configure the password for the key.

   NOTE
   Passwords are composed of hexadecimal characters 0 through 9 and a through f. When AES-128-CMAC is used as the authentication algorithm, 16 hexadecimal characters must be configured. When AES-256-CMAC is used, 32 hexadecimal characters must be configured.

   ```
   device(config-keychain-mka-sample-key-100)# password
   12345678123456781234567809abcdef12345678123456781234567809abcdef
   ```

5. Configure the send-lifetime start and end times.

   NOTE
   If you use the keywords **end infinite** as shown in the example instead of a specific end time, the key remains active indefinitely.

   ```
   device(config-keychain-mka-sample-key-100)# send-lifetime start 02-14-2022 01:01:01 end infinite
   ```

6. (Optional) Configure the local timezone (as configured in the system) to be used for the start and end timers. If not configured, the lifetime values are based on the GMT clock time.

   ```
   device(config-keychain-mka-sample-key-100)# send life-time local
   ```

7. (Optional) Configure the tolerance value for the keys.

   NOTE
   Because of the potential for key overlap when the duration between the first key end-time and the following key start-time is short, RUCKUS recommends that you configure a minimum tolerance of 180 seconds to maintain hitless key rollover.

   The following example configures a 200 second tolerance period for the keychain profile "mka-sample-key-100."

```
device(config-keychain-mka-sample-key-100)# tolerance 200
```

The following example creates the MKA keychain "sample" and configures the underlying options. The configured options are confirmed in the output of the **show keychain name** command.

```
device# configure terminal
device(config)# keychain sample mka
device(config-keychain-mka-sample)# key-id 100
device(config-keychain-mka-sample-key-100)# authentication-algorithm aes-128-cmac
device(config-keychain-mka-sample-key-100)# password
12345678123456781234567809abcdef12345678123456781234567809abcdef
device(config-keychain-mka-sample-key-100)# send-lifetime start 02-16-2022 04:05:00 end infinite
device(config-keychain-mka-sample-key-100)# tolerance 200
device(config-keychain-mka-sample-key-100)# end
device# show keychain name sample
Keychain: sample
  Tolerance: 0
  Key-id   : 100
      AuthAlgorithm: aes-128-cmac
      Key-String   : *******
      Send Lifetime:-
          Start    : 02-16-2022 04:05:00 End         : Infinite
          Active   : Yes                 TimeToExpire: Infinite
          Timezone : GMT+00
```

Next: Enable interfaces for MACsec, and apply the MKA group configuration and MKA keychain as described in Enabling and Configuring Group Interfaces for MACsec on page 153.



© 2023 CommScope, Inc. All rights reserved.
350 West Java Dr., Sunnyvale, CA 94089 USA
https://www.commscope.com