# Security Configuration Guide, Cisco IOS XE Bengaluru 17.6.x (Catalyst 9500 Switches)

**First Published:** 2021-07-31

**Americas Headquarters**
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

The documentation set for this product strives to use bias-free language. For purposes of this documentation set, bias-free is defined as language that does not imply discrimination based on age, disability, gender, racial identity, ethnic identity, sexual orientation, socioeconomic status, and intersectionality. Exceptions may be present in the documentation due to language that is hardcoded in the user interfaces of the product software, language used based on standards documentation, or language that is used by a referenced third-party product.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: https://www.cisco.com/c/en/us/about/legal/trademarks.html. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2021 Cisco Systems, Inc. All rights reserved.

# CONTENTS


**Full Cisco Trademarks with Software License** ?

CHAPTER 1

**Controlling Switch Access with Passwords and Privilege Levels** 1

Restrictions for Controlling Switch Access with Passwords and Privileges 1

Restrictions and Guidelines for Reversible Password Types 1

Restrictions and Guidelines for Irreversible Password Types 1

Information About Controlling Switch Access with Passwords and Privileges 2

Preventing Unauthorized Access 2

Default Password and Privilege Level Configuration 3

Additional Password Security 3

Password Recovery 4

Terminal Line Telnet Configuration 4

Username and Password Pairs 4

Privilege Levels 5

AES Password Encryption and Master Encryption Keys 5

How to Configure Switch Access with Passwords and Privileges 6

Setting or Changing a Static Enable Password 6

Protecting Enable and Enable Secret Passwords with Encryption 7

Disabling Password Recovery 9

Setting a Telnet Password for a Terminal Line 10

Configuring Username and Password Pairs 11

Setting the Privilege Level for a Command 12

Changing the Default Privilege Level for Lines 13

Logging in to and Exiting a Privilege Level 14

Configuring an Encrypted Preshared Key 15

Monitoring Switch Access with Passwords and Privileges 16

Configuration Examples for Switch Access with Passwords and Privilege Levels 16
Example: Setting or Changing a Static Enable Password 16
Example: Protecting Enable and Enable Secret Passwords with Encryption 16
Example: Setting a Telnet Password for a Terminal Line 17
Example: Setting the Privilege Level for a Command 17
Example: Configuring an Encrypted Preshared Key 17
Feature History for Controlling Switch Access with Passwords and Privileges 17

CHAPTER 2 **Configuring Login Block** 19
Information About Login Enhancements-Login Block 19
Login Enhancements-Login Block Overview 19
Protecting Against Denial of Service and Dictionary Login Attacks 19
Delays Between Successive Login Attempts 20
Login Shutdown If DoS Attacks Are Suspected 20
How to Configure Login Enhancements-Login Block 20
Configuring Login Parameters 20
Verifying Login Parameters 22
Configuration Examples for Login Enhancements-Login Block 24
Example: Configuring Login Parameters 24
Feature History for Login Enhancements-Login Block 24

CHAPTER 3 **Configuring Authentication** 25
Prerequisites for Configuring Authentication 25
Restrictions for Configuring Authentication 25
Information About Authentication 25
Named Method Lists for Authentication 25
Method Lists and Server Groups 26
Login Authentication Using AAA 27
Login Authentication Using Enable Password 27
Login Authentication Using Kerberos 27
Login Authentication Using Line Password 27
Login Authentication Using Local Password 28
Login Authentication Using Group RADIUS 28
Login Authentication Using Group TACACS 28

Login Authentication Using Group Name 29
PPP Authentication Using AAA 29
PPP Authentication Using Kerberos 29
PPP Authentication Using Local Password 29
PPP Authentication Using Group RADIUS 30
PPP Authentication Using Group TACACS 30
PPP Authentication Using Group Name 30
AAA Scalability for PPP Requests 31
ARAP Authentication Using AAA 31
ARAP Authentication Allowing Authorized Guest Logins 31
ARAP Authentication Allowing Guest Logins 31
ARAP Authentication Using Line Password 32
ARAP Authentication Using Local Password 32
ARAP Authentication Using Group RADIUS 32
ARAP Authentication Using Group TACACS 32
ARAP Authentication Using a Group Name 32
NASI Authentication Using AAA 33
NASI Authentication Using Enable Password 33
NASI Authentication Using Group RADIUS 33
NASI Authentication Using Group TACACS 33
NASI Authentication Using Line Password 34
NASI Authentication Using Local Password 34
NASI Authentication Using Group Name 34
Specifying the Amount of Time for Login Input 34
Password Protection at the Privileged Level 35
Changing the Text Displayed at the Password Prompt 35
Double Authentication of PPP Sessions 35
How Double Authentication Works 36
Accessing the User Profile After Double Authentication 37
CHAP or PAP Authentication 37
Enabling PPP Encapsulation 38
Enabling PAP or CHAP 38
Inbound and Outbound Authentication 39
Enabling Outbound PAP Authentication 39

Refusing PAP Authentication Requests 39
Creating a Common CHAP Password 40
Refusing CHAP Authentication Requests 40
Delaying CHAP Authentication Until Peer Authenticates 40
Using MS-CHAP 40
Domain Stripping 41
How to Configure Authentication 41
Configuring Login Authentication Using AAA 41
Configuring PPP Authentication Using AAA 43
Configuring ARAP Authentication Using AAA 45
Configuring NASI Authentication Using AAA 47
Preventing an Access Request with a Blank Username from Being Sent to the RADIUS Server 48
Configuring Message Banners for AAA Authentication 49
Configuring a Login Banner 49
Configuring a Failed-Login Banner 50
Configuring AAA Packet of Disconnect 51
Configuring Double Authentication 52
Enabling Automated Double Authentication 53
Configuring Domain Stripping at the Server Group Level 56
Configuring Non-AAA Authentication Methods 57
Configuring Line Password Protection 57
Establishing Username Authentication 58
Defining PPP Authentication Using MS-CHAP 59
Configuration Examples for Authentication 61
Example: Configuring Method Lists 61
Example: RADIUS Authentication 63
Example: TACACS Authentication 64
Example: Kerberos Authentication 65
Example: AAA Scalability 66
Example: Configuring Login and Failed-Login Banners for AAA Authentication 67
Example: AAA Packet of Disconnect Server Key 68
Example: Double Authentication 68
Example: Configuration of the Local Host for AAA with Double Authentication 68

Example: Configuration of the AAA Server for First-Stage PPP Authentication and Authorization 69
Example: Configuration of the AAA Server for Second-Stage Per-User Authentication and Authorization 69
Example: Complete Configuration with TACACS 70
Example: Automated Double Authentication 73
Feature History for Configuring Authentication 74

CHAPTER 4
**Configuring Authorization 77**
Prerequisites for Configuring Authorization 77
Information About Configuring Authorization 78
Named Method Lists for Authorization 78
AAA Authorization Methods 78
Authorization Methods 79
Method Lists and Server Groups 80
AAA Authorization Types 80
Authorization Types 80
Authorization Attribute-Value Pairs 81
How to Configure Authorization 81
Configuring AAA Authorization Using Named Method Lists 81
Disabling Authorization for Global Configuration Commands 83
Configuring Authorization for Reverse Telnet 83
Configuration Examples for Authorization 84
Example: TACACS Authorization 84
Example: RADIUS Authorization 85
Example: Reverse Telnet Authorization 85
Feature History for Configuring Authorization 87

CHAPTER 5
**Configuring Accounting 89**
Prerequisites for Configuring Accounting 89
Restrictions for Configuring Accounting 89
Information About Configuring Accounting 90
Named Method Lists for Accounting 90
Method Lists and Server Groups 91

AAA Accounting Methods 91
AAA Accounting Types 93
Network Accounting 93
EXEC Accounting 96
Command Accounting 97
Connection Accounting 98
System Accounting 99
Resource Accounting 100
AAA Accounting Enhancements 102
AAA Broadcast Accounting 102
AAA Session MIB 102
Accounting Attribute-Value Pairs 103
How to Configure AAA Accounting 103
Configuring AAA Accounting Using Named Method Lists 103
Suppressing Generation of Accounting Records for Null Username Sessions 104
Generating Interim Accounting Records 105
Configuring an Alternate Method to Enable Periodic Accounting Records 105
Generating Interim Service Accounting Records 106
Generating Accounting Records for a Failed Login or Session 107
Specifying Accounting NETWORK-Stop Records Before EXEC-Stop Records 108
Suppressing System Accounting Records over Switchover 108
Configuring AAA Resource Failure Stop Accounting 108
Configuring AAA Resource Accounting for Start-Stop Records 108
AAA Broadcast Accounting 109
Configuring Per-DNIS AAA Broadcast Accounting 109
Establishing a Session with a Device if the AAA Server Is Unreachable 109
Monitoring Accounting 110
Troubleshooting Accounting 110
Configuration Examples for AAA Accounting 110
Example: Configuring a Named Method List 110
Example: Configuring AAA Resource Accounting 112
Example: Configuring AAA Broadcast Accounting 113
Example: Configuring per-DNIS AAA Broadcast Accounting 113
Example: AAA Session MIB 114

Additional References for Configuring Accounting 114
Feature History for Configuring Accounting 115

CHAPTER 6 **Configuring Local Authentication and Authorization** 117
How to Configure Local Authentication and Authorization 117
Configuring the Switch for Local Authentication and Authorization 117
Monitoring Local Authentication and Authorization 119
Feature History for Local Authentication and Authorization 119

CHAPTER 7 **Configuring AAA Dead-Server Detection** 121
Prerequisites for AAA Dead-Server Detection 121
Restrictions for AAA Dead-Server Detection 121
Information About AAA Dead-Server Detection 121
Criteria for Marking a RADIUS Server As Dead 122
How to Configure AAA Dead-Server Detection 122
Configuring AAA Dead-Server Detection 122
Verifying AAA Dead-Server Detection 123
Configuration Examples for AAA Dead-Server Detection 124
Example: Configuring AAA Dead-Server Detection 124
Feature History for AAA Dead-Server Detection 125

CHAPTER 8 **Configuring TACACS+** 127
Prerequisites for TACACS+ 127
Information About TACACS+ 128
TACACS+ and Switch Access 128
TACACS+ Overview 128
TACACS+ Operation 129
Method List 130
TACACS+ Configuration Options 130
TACACS+ Login Authentication 131
TACACS+ Authorization for Privileged EXEC Access and Network Services 131
TACACS+ Accounting 131
Default TACACS+ Configuration 131
How to Configure TACACS+ 131

Identifying the TACACS+ Server Host and Setting the Authentication Key 131
Configuring TACACS+ Login Authentication 133
Configuring TACACS+ Authorization for Privileged EXEC Access and Network Services 135
Starting TACACS+ Accounting 136
Establishing a Session with a Device if the AAA Server is Unreachable 137
Configuring TACACS Source-Interface Under a TACACS Server-Group 137
Monitoring TACACS+ 139
Additional References for TACACS+ 139
Feature History for TACACS+ 140

CHAPTER 9 **Device Sensor** 141
Restrictions for Device Sensor 141
Information About Device Sensor 141
Device Sensor 141
How to Configure Device Sensor 143
Enabling Accounting Augmentation 143
Configuring Protocol Attributes in Access and Accounting Requests 144
Creating a Protocol Filter 145
Applying a Protocol Filter to the Sensor Output 146
Tracking TLV Changes 147
Verifying the Device Sensor Configuration 148
Configuration Examples for Device Sensor 149
Examples: Configuring the Device Sensor 149
Feature History for Device Sensor 150

CHAPTER 10 **Configuring RADIUS** 151
Prerequisites for Configuring RADIUS 151
Restrictions for Configuring RADIUS 152
Information about RADIUS 152
RADIUS and Switch Access 152
RADIUS Overview 152
RADIUS Operation 153
RADIUS Change of Authorization 154
Change-of-Authorization Requests 155

CoA Request Response Code 157
CoA Request Commands 158
Default RADIUS Configuration 160
RADIUS Server Host 160
RADIUS Login Authentication 161
AAA Server Groups 161
AAA Authorization 161
RADIUS Accounting 162
Vendor-Specific RADIUS Attributes 162
Vendor-Proprietary RADIUS Server Communication 173
DSCP marking for RADIUS packets 173
How to Configure RADIUS 174
Identifying the RADIUS Server Host 174
Configuring RADIUS Login Authentication 175
Defining AAA Server Groups 178
Configuring RADIUS Authorization for User Privileged Access and Network Services 179
Starting RADIUS Accounting 180
Configuring Settings for All RADIUS Servers 181
Configuring the Device to Use Vendor-Specific RADIUS Attributes 182
Configuring the Device for Vendor-Proprietary RADIUS Server Communication 182
Configuring DSCP Marking on a RADIUS Server 183
Configuring the Source Interface and DSCP Marking on RADIUS Server Group 185
Configuring CoA on the Device 186
Configuring RADIUS Source-Interface Under a RADIUS Server-Group 187
Monitoring CoA Functionality 189
Feature History for RADIUS 189

CHAPTER 11 **Configuring RadSec** 191
Restrictions for Configuring RadSec 191
Information About RadSec 192
How to Configure RadSec 192
Configuring RadSec over TLS 192
Configuring Dynamic Authorization for TLS CoA 194
Configuring RadSec over DTLS 195

Configuring Dynamic Authorization for DTLS CoA 196
Monitoring RadSec 197
Configuration Examples for RadSec 197
Example: Configuring RadSec over TLS 198
Example: Configuring Dynamic Authorization for TLS CoA 198
Example: Configuring RadSec over DTLS 198
Example: Configuring Dynamic Authorization for DTLS CoA 198
Feature History for Configuring RadSec 199

CHAPTER 12 **RADIUS Server Load Balancing** 201
Prerequisites for RADIUS Server Load Balancing 201
Restrictions for RADIUS Server Load Balancing 201
Information About RADIUS Server Load Balancing 202
RADIUS Server Load Balancing Overview 202
Transaction Load Balancing Across RADIUS Server Groups 202
RADIUS Server Status and Automated Testing 203
VRF-Aware RADIUS Automated Testing 203
How to Configure RADIUS Server Load Balancing 204
Enabling Load Balancing for a Named RADIUS Server Group 204
Troubleshooting RADIUS Server Load Balancing 205
Enabling VRF Aware RADIUS Automated Testing 206
Configuration Examples for RADIUS Server Load Balancing 207
Example: Enabling Load Balancing for a Named RADIUS Server Group 207
Example: Monitoring Idle Timer 210
Example: Configuring the Preferred Server with the Same Authentication and Authorization Server 210
Example: Configuring the Preferred Server with Different Authentication and Authorization Servers 211
Example: Configuring the Preferred Server with Overlapping Authentication and Authorization Servers 211
Example: Configuring the Preferred Server with Authentication Servers As a Subset of Authorization Servers 212
Example: Configuring the Preferred Server with Authentication Servers As a Superset of Authorization Servers 212
Example: Enabling VRF Aware RADIUS Automated Testing 213

Additional References for RADIUS Server Load Balancing 213
Feature History for RADIUS Server Load Balancing 213

CHAPTER 13 **MACsec Encryption** 215
Prerequisites for MACsec Encryption 215
Restrictions for MACsec Encryption 215
Information About MACsec Encryption 216
Recommendations for MACsec Encryption 216
MACsec Encryption Overview 217
Media Access Control Security and MACsec Key Agreement 217
MKA Policies 218
Definition of Policy-Map Actions 218
Virtual Ports 219
MKA Statistics 219
Key Lifetime and Hitless Key Rollover 219
Replay Protection Window Size 220
MACsec, MKA, and 802.1x Host Modes 220
Certificate-Based MACsec Encryption 221
MACsec Connections Across Intermediate Switches 221
Limitations for MACsec Connections Across Intermediate Switches 222
Switch-to-switch MKA MACsec Must Secure Policy 222
MACsec Extended Packet Numbering (XPN) 222
MKA/MACsec for Port Channel 223
MACsec Cipher Announcement 223
How to Configure MACsec Encryption 224
Configuring MKA and MACsec 224
Configuring an MKA Policy 224
Configuring Switch-to-host MACsec Encryption 225
Configuring MKA MACsec using PSK 228
Configuring MACsec MKA using PSK 228
Configuring MACsec MKA on an Interface using PSK 229
Configuring Certificate-Based MACsec Encryption 230
Generating Key Pairs 231
Configuring Enrollment using SCEP 231

Configuring Enrollment Manually 234
Configuring Switch-to-switch MACsec Encryption 236
Configuring MACsec XPN 237
Configuring an MKA Policy for XPN 237
Applying the XPN MKA Policy to an Interface 238
Configuring MKA/MACsec for Port Channel 239
Configuring MKA/MACsec for Port Channel using PSK 239
Configuring Port Channel Logical Interfaces for Layer 2 EtherChannels 241
Configuring Port Channel Logical Interfaces for Layer 3 EtherChannels 242
Configuring MACsec Cipher Announcement 243
Configuring an MKA Policy for Secure Announcement 243
Configuring Secure Announcement Globally 244
Configuring EAPOL Announcements on an Interface 244
Configuring Cisco TrustSec MACsec 245
Configuring Cisco TrustSec Switch-to-Switch Link Security in Manual Mode 245
Configuration Examples for MACsec Encryption 247
Example: Configuring MKA and MACsec 247
Example: Configuring MACsec MKA using PSK 248
Example: Configuring MACsec MKA Using Certificate-Based MACsec Encryption 249
Example: Configuring MACsec XPN 249
Example: Configuring MACsec MKA for Port Channel using PSK 252
Example: Configuring MACsec Cipher Announcement 258
Example: Displaying MKA Information 262
Additional References for MACsec Encryption 268
Feature History for MACsec Encryption 269

CHAPTER 14

**Configuring Secure Shell 271**
Configuring Secure Shell 271
Prerequisites for Configuring Secure Shell 271
Restrictions for Configuring Secure Shell 272
Information About Configuring Secure Shell 272
SSH Server 272
SSH Integrated Client 272
RSA Authentication Support 273

SSH Servers, Integrated Clients, and Supported Versions 273
SSH Configuration Guidelines 273
How to Configure Secure Shell 274
Setting Up the Device to Run SSH 274
Configuring an SSH Server 275
Invoking an SSH Client 276
Configuration Examples for Secure Shell 277
Example: Configuring an SSH Server 277
Example: Invoking an SSH Client 277
Example: Verifying SSH 277
Additional References for Secure Shell 278
Feature History for Configuring Secure Shell 278

CHAPTER 15 **Secure Shell Version 2 Support** 281
Prerequisites for Secure Shell Version 2 Support 281
Restrictions for Secure Shell Version 2 Support 282
Information About Secure Shell Version 2 Support 282
Secure Shell Version 2 282
Secure Shell Version 2 Enhancements 283
Secure Shell Version 2 Enhancements for RSA Keys 283
SSH And Switch Access 284
SNMP Trap Generation 284
SSH Keyboard Interactive Authentication 284
How to Configure Secure Shell Version 2 Support 285
Configuring a Device for SSH Version 2 Using a Hostname and Domain Name 285
Configuring a Device for SSH Version 2 Using RSA Key Pairs 286
Configuring the Cisco SSH Server to Perform RSA-Based User Authentication 287
Configuring the Cisco IOS SSH Client to Perform RSA-Based Server Authentication 288
Starting an Encrypted Session with a Remote Device 290
Verifying the Status of the Secure Shell Connection 291
Verifying the Secure Shell Version 2 Status 292
Monitoring and Maintaining Secure Shell Version 2 293
Configuration Examples for Secure Shell Version 2 Support 296
Example: Configuring Secure Shell Version 2 296

Example: Configuring Secure Shell Versions 1 and 2 296
Example: Starting an Encrypted Session with a Remote Device 296
Example: Setting an SNMP Trap 296
Examples: SSH Keyboard Interactive Authentication 296
Example: Enabling Client-Side Debugs 296
Example: Enabling ChPass with a Blank Password Change 297
Example: Enabling ChPass and Changing the Password on First Login 297
Example: Enabling ChPass and Expiring the Password After Three Logins 298
Example: SNMP Debugging 298
Examples: SSH Debugging Enhancements 299
Additional References for Secure Shell Version 2 Support 300
Feature History for Secure Shell Version 2 Support 301

CHAPTER 16 **SSH Support Over IPv6** 303
Prerequisites for SSH Support over IPv6 303
Information About SSH Support over IPv6 303
SSH over an IPv6 Transport 303
How to Configure SSH Support over IPv6 304
Enabling SSH on an IPv6 Device 304
Configuration Examples for SSH Support over IPv6 305
Example: Enabling SSH on an IPv6 Device 305
Additional References for SSH Support over IPv6 305
Feature History for SSH Support over IPv6 305

CHAPTER 17 **Configuring SSH File Transfer Protocol** 307
Prerequisites for SSH File Transfer Protocol 307
Restrictions for SSH File Transfer Protocol 307
Information About SSH Support over IPv6 308
SSH File Transfer Protocol Overview 308
How to Configure SSH File Transfer Protocol 308
Configuring SFTP 308
Performing an SFTP Copy Operation 309
Configuration Examples for SSH Support over IPv6 309
Example: Configuring SSH File Transfer Protocol 309

Additional References for SSH File Transfer Protocol 310
Feature History for SSH File Transfer Protocol 310

CHAPTER 18 **X.509v3 Certificates for SSH Authentication 311**
X.509v3 Certificates for SSH Authentication 311
Prerequisites for X.509v3 Certificates for SSH Authentication 311
Restrictions for X.509v3 Certificates for SSH Authentication 311
Information About X.509v3 Certificates for SSH Authentication 311
Digital Certificates 312
Server and User Authentication using X.509v3 312
How to Configure X.509v3 Certificates for SSH Authentication 312
Configuring the SSH Server to Use Digital Certificates for Server Authentication 312
Configuring the SSH Server to Verify Digital Certificates for User Authentication 313
Configuring Trustpoint Authentication and Creating Device Certificate 315
Verifying Configuration for Server and User Authentication Using Digital Certificates 317
Configuration Examples for X.509v3 Certificates for SSH Authentication 317
Example: Configuring the SSH Server to Use Digital Certificates for Server Authentication 317
Example: Configuring the SSH Server to Verify Digital Certificates for User Authentication 318
Feature History for X.509v3 Certificates for SSH Authentication 318

CHAPTER 19 **SSH Algorithms for Common Criteria Certification 319**
Restriction for SSH Algorithms for Common Criteria Certification 319
Information About SSH Algorithms for Common Criteria Certification 319
SSH Algorithms for Common Criteria Certification 319
Cisco IOS SSH Server Algorithms 319
Cisco IOS SSH Client Algorithms 320
How to Configure SSH Algorithms for Common Criteria Certification 321
Configuring an Encryption Key Algorithm for a Cisco IOS SSH Server and Client 321
Troubleshooting Tips 322
Configuring a MAC Algorithm for a Cisco IOS SSH Server and Client 322
Troubleshooting Tips 323
Configuring a Host Key Algorithm for a Cisco IOS SSH Server 323
Troubleshooting Tips 324
Configuration Examples For SSH Algorithms for Common Criteria Certification 325

Example: Configuring Encryption Key Algorithms for a Cisco IOS SSH Server 325
Example: Configuring Encryption Key Algorithms for a Cisco IOS SSH Client 325
Example: Configuring MAC Algorithms for a Cisco IOS SSH Server 325
Example: Configuring Host Key Algorithms for a Cisco IOS SSH Server 325
Verifying SSH Algorithms for Common Criteria Certification 326
Feature History for Secure Shell Algorithms for Common Criteria Certification 327

CHAPTER 20 **Configuring Secure Socket Layer HTTP** 329
Information About Secure Socket Layer HTTP 329
Secure HTTP Servers and Clients Overview 329
Certificate Authority Trustpoints 329
CipherSuites 331
Default SSL Configuration 332
SSL Configuration Guidelines 332
How to Configure Secure Socket Layer HTTP 332
Configuring a CA Trustpoint 332
Configuring the Secure HTTP Server 334
Configuring the Secure HTTP Client 337
Monitoring Secure HTTP Server and Client Status 338
Additional References for Secure Socket Layer HTTP 339
Feature History for Secure Socket Layer HTTP 339

CHAPTER 21 **IPv4 ACLs** 341
Restrictions for IPv4 Access Control Lists 341
Information About IPv4 Access Control Lists 342
ACL Overview 342
Access Control Entries 343
ACL Supported Types 343
Supported ACLs 343
ACL Precedence 343
Port ACLs 344
Router ACLs 345
VLAN Maps 346
ACEs and Fragmented and Unfragmented Traffic 346

Standard and Extended IPv4 ACLs 347
IPv4 ACL Switch Unsupported Features 347
Access List Numbers 347
Numbered Standard IPv4 ACLs 348
Numbered Extended IPv4 ACLs 348
Named IPv4 ACLs 349
ACL Logging 349
Hardware and Software Treatment of IP ACLs 350
VLAN Map Configuration Guidelines 351
VLAN Maps with Router ACLs 351
VLAN Maps and Router ACL Configuration Guidelines 351
Time Ranges for ACLs 352
IPv4 ACL Interface Considerations 353
How to Configure IPv4 Access Control Lists 353
Configuring IPv4 ACLs 353
Creating a Numbered Standard ACL 353
Creating a Numbered Extended ACL 354
Creating Named Standard ACLs 357
Creating Extended Named ACLs 358
Configuring Time Ranges for ACLs 359
Applying an IPv4 ACL to a Terminal Line 360
Applying an IPv4 ACL to an Interface 361
Creating Named MAC Extended ACLs 362
Applying a MAC ACL to a Layer 2 Interface 363
Configuring an IPv4 ACL in Template Mode 364
Configuring VLAN Maps 366
Applying a VLAN Map to a VLAN 368
Monitoring IPv4 ACLs 369
Configuration Examples for IPv4 Access Control Lists 369
ACLs in a Small Networked Office 369
Examples: ACLs in a Small Networked Office 370
Example: Numbered ACLs 371
Examples: Extended ACLs 371
Examples: Named ACLs 372

Examples: ACL Logging 373
Example: ACEs and Fragmented and Unfragmented Traffic 374
Examples: Using Time Ranges with ACLs 375
Examples: Time Range Applied to an IP ACL 376
Examples: Including Comments in ACLs 376
Example: Creating an ACL and a VLAN Map to Deny a Packet 377
Example: Creating an ACL and a VLAN Map to Permit a Packet 377
Example: Default Action of Dropping IP Packets and Forwarding MAC Packets 377
Example: Default Action of Dropping MAC Packets and Forwarding IP Packets 378
Example: Default Action of Dropping All Packets 378
Example: Using VLAN Maps in a Network 379
Example: Wiring Closet Configuration 379
Example: Restricting Access to a Server on Another VLAN 380
Example: Denying Access to a Server on Another VLAN 380
Additional References for IPv4 Access Control Lists 381
Feature History for IPv4 Access Control Lists 381

CHAPTER 22 **IPv6 ACLs** 385
Restrictions for IPv6 ACLs 385
Information About IPv6 ACLs 386
IPv6 ACL Overview 386
Supported ACLs 386
Types of ACL 386
Per-User IPv6 ACL 386
Filter ID IPv6 ACL 387
Downloadable IPv6 ACL 387
ACL Precedence 387
VLAN Maps 387
Interactions with Other Features and Switches 388
How to Configure an IPv6 ACL 388
Default Configuration for IPv6 ACLs 388
Configuring IPv6 ACLs 389
Attaching an IPv6 ACL to an Interface 391
Configuring an IPv6 ACL in Template Mode 392

Configuring a VLAN Map 394
Applying a VLAN Map to a VLAN 396
Monitoring IPv6 ACLs 397
Configuration Examples for IPv6 ACL 397
Example: Creating an IPv6 ACL 397
Example: Displaying IPv6 ACLs 397
Example: Displaying VLAN Access Map Configuration 398
Feature History for IPv6 ACLs 398

CHAPTER 23 **Object Groups for ACLs** 401
Object Groups for ACLs 401
Restrictions for Object Groups for ACLs 401
Information About Object Groups for ACLs 401
Object Groups 402
ACLs Based on Object Groups 403
How to Configure Object Groups for ACLs 403
Creating a Network Object Group 403
Creating a Service Object Group 405
Creating an Object-Group-Based ACL 406
Applying an Object Group-Based ACL to an Interface 409
Verifying Object Groups for ACLs 410
Configuration Examples for Object Groups for ACLs 410
Example: Creating a Network Object Group 410
Example: Creating a Service Object Group 411
Example: Creating an Object Group-Based ACL 411
Applying an Object Group-Based ACL to an Interface 411
Example: Verifying Object Groups for ACLs 412
Additional References for Object Groups for ACLs 413
Feature History for Object Groups for ACLs 413

CHAPTER 24 **Configuring IP Source Guard** 415
Information About IP Source Guard 415
IP Source Guard 415
IP Source Guard for Static Hosts 415

IP Source Guard Configuration Guidelines 416
How to Configure IP Source Guard 417
Enabling IP Source Guard 417
Configuring IP Source Guard for Static Hosts on a Layer 2 Access Port 418
Monitoring IP Source Guard 419
Feature History for IP Source Guard 420

CHAPTER 25 **Configuring Dynamic ARP Inspection** 421
Restrictions for Dynamic ARP Inspection 421
Information About Dynamic ARP Inspection 422
Understanding Dynamic ARP Inspection 422
Interface Trust States and Network Security 424
Rate Limiting of ARP Packets 425
Relative Priority of ARP ACLs and DHCP Snooping Entries 425
Logging of Dropped Packets 425
Default Dynamic ARP Inspection Configuration 425
Relative Priority of ARP ACLs and DHCP Snooping Entries 426
How to Configure Dynamic ARP Inspection 426
Configuring ARP ACLs for Non-DHCP Environments 426
Configuring Dynamic ARP Inspection in DHCP Environments 429
Limiting the Rate of Incoming ARP Packets 431
Performing Dynamic ARP Inspection Validation Checks 432
Monitoring DAI 434
Verifying the DAI Configuration 434
Feature History for Dynamic ARP Inspection 434

CHAPTER 26 **Configuring IPv6 First Hop Security** 437
Prerequisites for IPv6 First Hop Security 437
Restrictions for IPv6 First Hop Security 437
Information About IPv6 First Hop Security 437
How to Configure IPv6 First Hop Security 441
Configuring an IPv6 Snooping Policy 441
Attaching an IPv6 Snooping Policy to an Interface 443
Attaching an IPv6 Snooping Policy to a Layer 2 EtherChannel Interface 444

Attaching an IPv6 Snooping Policy to VLANs Globally 445
Configuring the IPv6 Binding Table Content 446
Configuring an IPv6 Neighbor Discovery Inspection Policy 447
Attaching an IPv6 Neighbor Discovery Inspection Policy to an Interface 449
Attaching an IPv6 Neighbor Discovery Inspection Policy to a Layer 2 EtherChannel Interface 450
Attaching an IPv6 Neighbor Discovery Inspection Policy to VLANs Globally 451
Configuring an IPv6 Router Advertisement Guard Policy 452
Attaching an IPv6 Router Advertisement Guard Policy to an Interface 454
Attaching an IPv6 Router Advertisement Guard Policy to a Layer 2 EtherChannel Interface 455
Attaching an IPv6 Router Advertisement Guard Policy to VLANs Globally 456
Configuring an IPv6 DHCP Guard Policy 457
Attaching an IPv6 DHCP Guard Policy to an Interface or a VLAN on an Interface 458
Attaching an IPv6 DHCP Guard Policy to a Layer 2 EtherChannel Interface 459
Attaching an IPv6 DHCP Guard Policy to VLANs Globally 460
Configuring IPv6 Source Guard 461
Attaching an IPv6 Source Guard Policy to an Interface 462
Attaching an IPv6 Source Guard Policy to a Layer 2 EtherChannel Interface 463
Configuring IPv6 Prefix Guard 464
Attaching an IPv6 Prefix Guard Policy to an Interface 465
Attaching an IPv6 Prefix Guard Policy to a Layer 2 EtherChannel Interface 465
Configuring an IPv6 Destination Guard Policy 466
Configuration Examples for IPv6 First Hop Security 467
Example: Configuring an IPv6 DHCP Guard Policy 467
Examples: Attaching an IPv6 Source Guard Policy to a Layer 2 EtherChannel Interface 468
Examples: Attaching an IPv6 Prefix Guard Policy to a Layer 2 EtherChannel Interface 468
Example: Using the Data-Glean Recovery Function 468
Additional References for IPv6 First Hop Security 471
Feature History for IPv6 First Hop Security 471

CHAPTER 27 **Configuring Switch Integrated Security Features** 473
Information About SISF 473
Overview 473
Understanding the SISF Infrastructure 474
The Binding Table 475

States and Lifetime of a Binding Table Entry 476
Binding Table Sources 478
Device-Tracking 480
Actions on a Packet and Actions on the Binding Table 480
Device-Tracking Policy 481
Understanding Policy Parameters 481
Glean versus Guard versus Inspect 481
Trusted-Port and Device-Role Switch 483
Address Count Limits 491
Tracking 493
Guidelines for Policy Creation 493
Guidelines for Applying a Policy 493
How to Configure SISF 494
Applying the Default Device Tracking Policy to a Target 495
Creating a Custom Device Tracking Policy with Custom Settings 496
Attaching a Device Tracking Policy to an Interface 500
Attaching a Device Tracking Policy to a VLAN 501
Using an Interface Template to Enable SISF 502
Migrating from Legacy IPDT and IPv6 Snooping to SISF-Based Device Tracking 503
Configuration Examples for SISF 504
Example: Programatically Enabling SISF by Configuring DHCP Snooping 504
Example: Programatically Enabling SISF by Configuring EVPN on VLAN 505
Example: Programatically Enabling SISF by Configuring LISP (LISP-DT-GLEAN-VLAN) 505
Example: Programatically enabling SISF by Configuring LISP (LISP-DT-GUARD-VLAN) 506
Example: Programatically Enabling SISF by Configuring LISP (LISP-DT-GUARD-VLAN-MULTI-IP) 507
Example: Mitigating the IPv4 Duplicate Address Problem 508
Example: Disabling IPv6 Device Tracking on a Target 509
Example: Enabling IPv6 for SVI on VLAN (To Mitigate the Duplicate Address Problem) 510
Example: Configuring a Multi-Switch Network to Stop Creating Binding Entries from a Trunk Port 510
Example: Avoiding a Short Device-Tracking Binding Reachable Time 511
Feature History for SISF 511

CHAPTER 28 **Configuring IEEE 802.1x Port-Based Authentication** 513

Restrictions for IEEE 802.1x Port-Based Authentication 513

Information About IEEE 802.1x Port-Based Authentication 514

Overview of IEEE 802.1x Port-Based Authentication 514

Port-Based Authentication Process 515

Port-Based Authentication Initiation and Message Exchange 517

Port-Based Authentication Methods 519

Per-User ACLs and Filter IDs 519

Ports in Authorized and Unauthorized States 520

802.1x Host Mode 521

Access Session Limit Profile 521

MAC Move 521

MAC Replace 522

802.1x Accounting 523

802.1x Accounting Attribute-Value Pairs 523

802.1x Readiness Check 524

Switch-to-RADIUS Server Communication 524

IEEE 802.1x Authentication 524

802.1x Authentication 524

Port-Based Authentication Manager CLI Commands 525

Default 802.1x Authentication Configuration 526

Port-Based Authentication and Switch Stacks 527

802.1x Authentication with VLAN Assignment 528

802.1x Authentication with Per-User ACLs 529

802.1x Authentication with Downloadable ACLs and Redirect URLs 530

VLAN ID-Based MAC Authentication 531

IEEE 802.1x Authentication with MAC Authentication Bypass 531

802.1x Multiple Authentication Mode 533

802.1x Authentication with Guest VLAN 535

802.1x Authentication with Restricted VLAN 536

802.1x Authentication with Inaccessible Authentication Bypass 536

VLAN Assignment, Guest VLAN, Restricted VLAN, and Inaccessible Authentication Bypass 538

802.1x Critical Voice VLAN 539

IEEE 802.1x Authentication with Voice VLAN Ports 539
IEEE 802.1x Authentication with Wake-on-LAN 540
Flexible Authentication Ordering 540
Open1x Authentication 541
Multidomain Authentication 541
802.1x Supplicant and Authenticator Switches with Network Edge Access Topology 543
802.1x User Distribution 544
802.1x User Distribution Configuration Guidelines 545
Network Admission Control Layer 2 IEEE 802.1x Validation 545
Voice Aware 802.1x Security 546
Common Session ID 546
Maximum Number of Allowed Devices Per Port 546
How to Configure IEEE 802.1x Port-Based Authentication 547
Configuring 802.1x Authentication 547
Configuring 802.1x Port-Based Authentication 547
Configuring Periodic Reauthentication 550
Configuring 802.1x Violation Modes 551
Changing the Quiet Period 553
Changing the Switch-to-Client Retransmission Time 554
Setting the Switch-to-Client Frame-Retransmission Number 555
Configuring Host Mode 556
Enabling MAC Move 558
Disabling MAC Move 559
Enabling MAC Replace 559
Configuring 802.1x Accounting 560
Configuring 802.1x Readiness Check 562
Configuring Switch-to-RADIUS Server Communication 563
Setting the Reauthentication Number 565
Configuring a Guest VLAN 566
Configuring a Restricted VLAN 567
Configuring the Number of Authentication Attempts on a Restricted VLAN 568
Configuring 802.1x Inaccessible Authentication Bypass with Critical Voice VLAN 569
Configuring 802.1x Authentication with Wake-on-LAN 572
Configuring MAC Authentication Bypass 573

Configuring 802.1x User Distribution 574
Configuring NAC Layer 2 802.1x Validation 575
Configuring an Authenticator Switch with NEAT 577
Configuring a Supplicant Switch with NEAT 578
Configuring 802.1x Authentication with Downloadable ACLs and Redirect URLs 580
Configuring Downloadable ACLs 581
Configuring a Downloadable Policy 582
Configuring VLAN ID Based MAC Authentication 584
Configuring Flexible Authentication Ordering 585
Configuring Open1x 586
Disabling 802.1x Authentication on a Port 588
Resetting the 802.1x Authentication Configuration to Default Values 589
Configuring Voice-Aware 802.1x Security 590
Configuration Examples for IEEE 802.1x Port-Based Authentication 592
Example: Configuring Inaccessible Authentication Bypass 592
Example: Configuring VLAN Groups 592
Monitoring IEEE 802.1x Port-Based Authentication Statistics and Status 593
Feature History for IEEE 802.1x Port-Based Authentication 594

CHAPTER 29 **IEEE 802.1X VLAN Assignment** 595
Prerequisites for IEEE 802.1X VLAN Assignment 595
Restrictions for IEEE 802.1X VLAN Assignment 596
Information About IEEE 802.1X VLAN Assignment 597
Configuring Authorization 597
IEEE 802.1X Authentication with VLAN Assignment 597
How to Configure IEEE 802.1X VLAN Assignment 597
Enabling AAA Authorization for VLAN Assignment 597
Enabling IEEE 802.1X Authentication and Authorization 598
Specifying an Authorized VLAN in the RADIUS Server Database 600
Configuration Examples for IEEE 802.1X VLAN Assignment 601
Example: Enabling AAA Authorization for VLAN Assignment 601
Example: Enabling 802.1X Authentication 601
Additional References for IEEE 802.1X Port-Based Authentication 602
Feature History for IEEE 802.1X VLAN Assignment 602

CHAPTER 30 **Web-Based Authentication** 605

Restrictions for Web-Based Authentication 605

Information About Web-Based Authentication 605

Web-Based Authentication Overview 605

Device Roles 606

Host Detection 607

Session Creation 607

Authentication Process 607

Local Web Authentication Banner 608

Web Authentication Customizable Web Pages 611

Guidelines 611

Authentication Proxy Web Page Guidelines 612

Redirection URL for Successful Login Guidelines 613

Web-based Authentication Interactions with Other Features 613

Port Security 613

LAN Port IP 613

Gateway IP 613

ACLs 613

EtherChannel 614

How to Configure Web-Based Authentication 614

Default Web-Based Authentication Configuration 614

Web-Based Authentication Configuration Guidelines and Restrictions 614

Configuring the Authentication Rule and Interfaces 616

Configuring AAA Authentication 617

Configuring Switch-to-RADIUS-Server Communication 619

Configuring the HTTP Server 621

Customizing the Authentication Proxy Web Pages 622

Specifying a Redirection URL for a Successful Login 623

Configuring Web-Based Authentication Parameters 624

Configuring a Web-Based Authentication Local Banner 624

Removing Web-Based Authentication Cache Entries 625

Verifying Web-Based Authentication 626

Feature History for Web-Based Authentication 626

CHAPTER 31 **Identity Based Networking Services Overview** 627

Information About Identity-Based Networking Services 627

Understanding Cisco Identity Based Networking Services 627

Features in Cisco Identity Based Networking Services 627

Benefits of Cisco Identity Based Networking Services 628

Web Authentication Support for Common Session ID 628

Web Authentication Support of IPv6 628

IP Device Tracking 629

Feature History for Cisco Identity Based Networking Services Overview 629

CHAPTER 32 **Change of Authorization Support** 631

Information About CoA Support 631

RADIUS Change-of-Authorization Support 631

Session Identification 632

CoA Activate Service Command 633

CoA Deactivate Service Command 633

CoA Bounce Host Port Command 633

CoA Disable Host Port Command 634

CoA Session Query Command 634

CoA Session Reauthenticate Command 634

CoA Session Terminate Command 635

Feature History for CoA Support 635

CHAPTER 33 **Configuring Identity Control Policies** 637

Information About Identity Control Policies 637

Cisco Identity Based Networking Services Configuration 637

Concurrent Authentication Methods 637

Configuration Display Mode 638

Control Policies for Cisco Identity Based Networking Services 638

Control Policy Configuration Overview 639

Parameter Maps for Cisco Identity Based Networking Services 639

Per User Inactivity Handling Across Methods 640

How to Configure Identity Control Policies 640

Enabling the Display Mode for Cisco Identity Based Networking Services 640
Configuring a Control Class 641
Configuring a Control Policy 644
Applying a Control Policy to an Interface 648
Configuring Authentication Features on Ports 649
Configuring a Parameter Map for Web-Based Authentication 650
Configuration Examples for Cisco Identity-Based Control Policies 653
Example: Configuring Control Policy for Concurrent Authentication Methods 653
Example: Configuring Control Policy for Sequential Authentication Methods 654
Example: Configuring Parameter Maps 656
Feature History for Identity Control Policies 657

CHAPTER 34 **Policy Classification Engine** 659
Restrictions for Policy Classification Engine 659
Information About Policy Classification Engine 659
Policy Classification Engine Overview 659
How to Configure Policy Classification Engine 660
Configuring Policies in Cisco Identity Based Networking Services 660
Configuring a Subscriber Parameter Map 660
Configuring a Subscriber Policy Map 662
Configuration Examples for Policy Classification Engine 663
Example: Configuring a Subscriber Parameter Map 663
Example: Configuring a Subscriber Policy Map 663
Feature History for Policy Classification Engine 663

CHAPTER 35 **Configuring Identity Service Templates** 665
Prerequisites for Identity Service Templates 665
Information About Identity Service Templates 665
Service Templates for Cisco Identity-Based Networking Services 665
Downloadable Service Templates 666
Locally Configured Service Templates 666
How to Configure Identity Service Templates 666
Configuring a Local Service Template 666
Configuration Examples for Identity Service Templates 668

Example: Activating a Service Template and Replace All 668
Example: Activating a Service Template for Fallback Service 669
Example: Deactivating a Service Template 669
Feature History for Identity Service Templates 670

CHAPTER 36 **Interface Templates** 673
Restrictions for Interface Templates 673
Information About Interface Templates 673
About Interface Templates 673
Binding an Interface Template to a Target 676
Priority for Configurations Using Interface Templates 676
How to Configure Interface Templates 677
Configuring Interface Templates 677
Configuring Static Binding for Interface Templates 678
Configuring Dynamic Binding of Interface Templates 679
Verifying an Interface Template 680
Configuration Examples for Interface Templates 686
Example: Configuring User Interface Templates 686
Example: Sourcing Interface Templates 686
Example: Dynamically Binding Interface Templates 687
Feature History for Interface Templates 687

CHAPTER 37 **Autoconf** 689
Prerequisites for Autoconf 689
Restrictions for Autoconf 689
Information About Autoconf 690
Benefits of Autoconf 690
Identity Session Management and Templates 690
Autoconf Operation 691
Advantages of Using Templates 693
Autoconf Functionality 694
How to Configure Autoconf 695
Applying a Built-in Template to an End Device 695
Applying a Modified Built-in Template to an End Device 701

Migrating from ASP to Autoconf 703
Configuration Examples for Autoconf 704
Example: Applying a Built-in Template to an End Device 704
Example: Applying a Modified Built-in Template to an End Device 704
Example: Migrating from ASP Macros to Autoconf 704
Feature History for Autoconf 705

CHAPTER 38 **eEdge Integration with MACsec 707**
Prerequisites for eEdge Integration with MACsec 707
Restrictions for eEdge Integration with MACsec 707
Information About eEdge Integration with MACsec 708
Overview of MACsec 708
MACsec Standard Encryption 708
EAP Implementation of MKA 708
eEdge Integration with MACsec 709
How to Configure eEdge Integration with MACsec 709
Integrating eEdge with MACsec 709
Identifying Link Layer Security Failures 710
Configuration Examples for eEdge Integration with MACsec 711
Example: Integrating eEdge with MACsec 711
Example: Identifying Linksec Failures 712
Feature History for eEdge Integration with MACsec 712

CHAPTER 39 **Critical Voice VLAN Support 713**
Restrictions for Critical Voice VLAN Support 713
Information About Critical Voice VLAN Support 713
Critical Voice VLAN Support in Multidomain Authentication Mode 713
Critical Voice VLAN Support in Multiauthentication Mode 714
Critical Voice VLAN Support in a Service Template 714
How to Configure Critical Voice VLAN Support 714
Configuring a Critical Voice VLAN in a Service Template 714
Activating Critical Voice VLAN 717
Configuration Examples for Critical Voice VLAN Support 719
Example: Configuring a Voice VLAN in a Service Template 719

Example: Activating a Critical Voice VLAN on a Service Template 719
Feature History for Critical Voice VLAN Support 719

CHAPTER 40 **Configuring Local Authentication Using LDAP** 721
Information About Local Authentication Using LDAP 721
Local Authentication Using LDAP 721
How to Configure Local Authentication Using LDAP 721
Configuring Local Authentication Using LDAP 721
Configuring MAC Filtering Support 722
Configuration Examples for Local Authentication Using LDAP 723
Example: Configuring Local Authentication Using LDAP 723
Example: Configuring MAC Filtering Support 724
Feature History for Local Authentication Using LDAP 724

CHAPTER 41 **Web Authentication Redirection to Original URL** 725
Web Authentication Redirection to Original URL Overview 725
Feature History for Local Authentication Using LDAP 727

CHAPTER 42 **Configuring Port-Based Traffic Control** 729
Port-Based Traffic Control 729
Information About Port-Based Traffic Control 729
Storm Control 729
Protected Ports 731
Port Blocking 731
How to Configure Port-Based Traffic Control 732
Configuring Storm Control and Threshold Levels 732
Configuring a Protected Port 734
Blocking Flooded Traffic on an Interface 735
Additional References for Port-Based Traffic Control 736
Feature History for Port-Based Traffic Control 737

CHAPTER 43 **Port Security** 739
Prerequisites for Port Security 739
Restrictions for Port Security 739

Information About Port Security 740
Port Security 740
Types of Secure MAC Addresses 740
Default MAC Address Table Settings 740
MAC Address Table Creation 740
Sticky Secure MAC Addresses 741
Security Violations 741
Port Security Aging 742
Default Port Security Configuration 743
Port Security Configuration Guidelines 743
How to Configure Port Security 744
Enabling and Configuring Port Security 744
Enabling and Configuring Port Security Aging 749
Changing the Address Aging Time 750
Monitoring Port Security 751
Configuration Examples for Port Security 752
Feature History for Port Security 752

CHAPTER 44 **Configuring Control Plane Policing** 755
Restrictions for CoPP 755
Information About CoPP 756
CoPP Overview 756
System-Defined Aspects of CoPP 756
User-Configurable Aspects of CoPP 762
Upgrading or Downgrading the Software Version 763
Software Version Upgrades and CoPP 763
Software Version Downgrades and CoPP 764
How to Configure CoPP 765
Enabling a CPU Queue or Changing the Policer Rate 765
Disabling a CPU Queue 766
Setting the Default Policer Rates for All CPU Queues 767
Configuration Examples for CoPP 768
Example: Enabling a CPU Queue or Changing the Policer Rate of a CPU Queue 768
Example: Disabling a CPU Queue 769

Example: Setting the Default Policer Rates for All CPU Queues 770
Monitoring CoPP 773
Feature History for CoPP 773

CHAPTER 45 **Configuring Lawful Intercept** 777
Prerequisites for Lawful Intercept 777
Restrictions for Lawful Intercept 777
Information About Lawful Intercept 778
Overview of Lawful Intercept 778
Benefits of Lawful Intercept 778
CALEA for Voice 779
Configuration Guidelines 779
Network Components Used for Lawful Intercept 780
Lawful Intercept Processing 781
Lawful Intercept MIBs 782
MIB Guidelines 783
Security Considerations 784
Restricting Access to the Lawful Intercept MIBs 784
How to Configure Lawful Intercept 785
Creating a Restricted SNMP View of Lawful Intercept MIBs 785
Enabling SNMP Notifications for Lawful Intercept 787
Disabling SNMP Notifications 788
Configuration Examples for Lawful Intercept 788
Example: Enabling Mediation Device Access Lawful Intercept MIBs 788
Feature History for Lawful Intercept 789

CHAPTER 46 **Configuring Authorization and Revocation of Certificates in a PKI** 791
Configuring Authorization and Revocation of Certificates in a PKI 791
Prerequisites for Authorization and Revocation of Certificates 791
Restrictions for Authorization and Revocation of Certificates 792
Information About Authorization and Revocation of Certificates 792
PKI Authorization 792
PKI and AAA Server Integration for Certificate Status 792
CRLs or OCSP Server Choosing a Certificate Revocation Mechanism 794

When to Use Certificate-Based ACLs for Authorization or Revocation 796
PKI Certificate Chain Validation 798
How to Configure Authorization and Revocation of Certificates in a PKI 798
Configuring PKI Integration with a AAA Server 798
Configuring a Revocation Mechanism for PKI Certificate Status Checking 803
Configuring Certificate Authorization and Revocation Settings 805
Configuring Certificate Chain Validation 813
Configuration Examples for Authorization and Revocation of Certificates in a PKI 814
Configuration and Verification Examples fo PKI AAA Authorization 814
Examples: Configuring a Revocation Mechanism 818
Example:Configuring a Hub Device at a Central Site for Certificate Revocation Checks 819
Examples:Configuring Certificate Authorization and Revocation Settings 823
Examples: Configuring Certificate Chain Validation 826
Feature History for Authorization and Revocation of Certificates in a PKI 827

CHAPTER 47

**Source Interface Selection for Outgoing Traffic with Certificate Authority 829**
Information About Source Interface Selection for Outgoing Traffic with Certificate Authority 829
Certificates That Identify an Entity 829
Source Interface for Outgoing TCP Connections Associated with a Trustpoint 830
How to Configure Source Interface Selection for Outgoing Traffic with Certificate Authority 830
Configuring the Interface for All Outgoing TCP Connections Associated with a Trustpoint 830
Configuration Examples for Source Interface Selection for Outgoing Traffic with Certificate Authority 833
Example:Source Interface Selection for Outgoing Traffic with Certificate Authority 833
Feature History for Source Interface Selection for Outgoing Traffic with Certificate Authority 833

CHAPTER 48

**Source Interface and VRF Support in LDAP 835**
Information About Source Interface and VRF Support in LDAP 835
Source Interface and VRF Support in LDAP Overview 835
How to Configure Source Interface and VRF Support in LDAP 836
Configuring LDAP Source Interface and VRF 836
Configuration Examples for Source Interface and VRF Support in LDAP 837
Example: Configuring LDAP Source Interface and VRF 837
Feature History for Source Interface and VRF Support in LDAP 837

CHAPTER 49 **Configuring IPv6 Support for LDAP** 839

Restrictions for Configuring IPv6 Support for LDAP 839

Information About Configuring IPv6 Support for LDAP 839

IPv6 Support for LDAP 839

Transport Layer Security 839

LDAP Operations 840

Bind 840

Compare 840

Search 840

How to Configure IPv6 Support for LDAP 841

Configuring Device-to-LDAP Server Communication 841

Configuring LDAP Protocol Parameters 842

Configuring Search and Bind Operations for an Authentication Request 843

Monitoring and Maintaining LDAP Scalability Enhancements 844

Configuration Examples of IPv6 Support for LDAP 846

Example: Device-to-LDAP Server Communication 846

Example: LDAP Protocol Parameters 846

Example: Search and Bind Operations for an Authentication Request 846

Example: Server Information from an LDAP Server 847

Additional References 847

Feature History for IPv6 Support for LDAP 848

CHAPTER 50 **Secure Operation in FIPS Mode** 849

FIPS 140-2 Overview 849

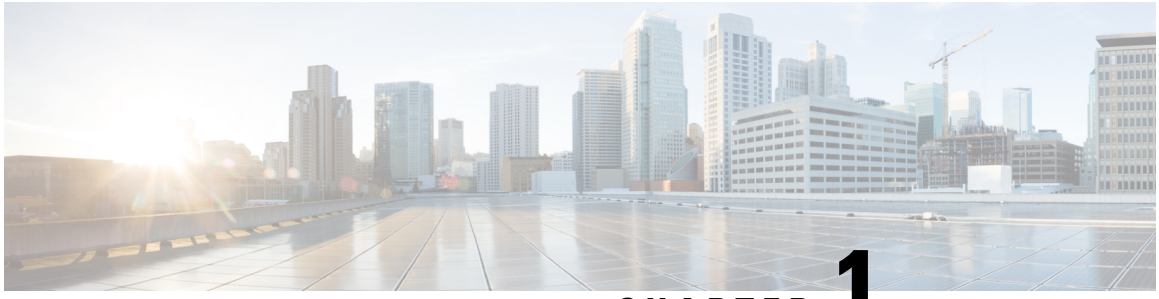
Configure FIPS 140-2 850

Key Zeroization 850

Disable FIPS Mode 851

Verify FIPS Configuration 851

Additional References for Secure Operation in FIPS Mode 852



CHAPTER 1

# Controlling Switch Access with Passwords and Privilege Levels


- Restrictions for Controlling Switch Access with Passwords and Privileges, on page 1
- Information About Controlling Switch Access with Passwords and Privileges, on page 2
- How to Configure Switch Access with Passwords and Privileges, on page 6
- Monitoring Switch Access with Passwords and Privileges, on page 16
- Configuration Examples for Switch Access with Passwords and Privilege Levels, on page 16
- Feature History for Controlling Switch Access with Passwords and Privileges, on page 17


# Restrictions for Controlling Switch Access with Passwords and Privileges

The following are the restrictions for controlling switch access with passwords and privileges:

- Disabling password recovery will not work if you have set the switch to boot up manually by using the **boot manual** global configuration command. This command produces the boot loader prompt (*switch:*) after the switch is power cycled.

## Restrictions and Guidelines for Reversible Password Types

- Password type 0 and 7 are replaced with password type 6. So password type 0 and 7, which were used for administrator login to the console, Telnet, SSH, webUI, and NETCONF must be migrated to password type 6. No action is required if username and password are type 0 and 7 for local authentication such as CHAP, EAP, and so on.
- If the startup configuration has a type 6 password and you downgrade to a version in which type 6 password is not supported, you can/may be locked out of the device.

## Restrictions and Guidelines for Irreversible Password Types

- Username secret password type 5 and enable secret password type 5 must be migrated to the stronger password type 8 or 9. For more information, see Protecting Enable and Enable Secret Passwords with Encryption, on page 7.

- If the startup configuration of the device has convoluted type 9 secret (password that starts with $14$), then a downgrade can only be performed to a release in which the convoluted type 9 secret is supported. Convoluted type 9 secret is supported in Cisco IOS XE Gibraltar 16.11.2 and later releases. If the startup configuration has a convoluted type 9 secret, and you downgrade to a release prior to Cisco IOS XE Gibraltar 16.11.2, you can/may be locked out of the device.

  Before you downgrade to any release in which convoluted type 9 secret is not supported, ensure that the type 9 secret (password that starts with $9$) must be part of the startup configuration instead of convoluted type 9 secret (password that starts with $14$) or type 5 secret (password that starts with $1$).

  If a device is upgraded from Cisco IOS XE Fuji 16.9.x, Cisco IOS XE Gibraltar 16.10.x, or Cisco IOS XE Gibraltar 16.11.x to Cisco IOS XE Gibraltar 16.12.x, the type 5 secret is auto-converted to convoluted type 9 secret (password that starts with $14$). For example: `username user1 secret 5 $1$dNmW$7jWhqdtZ2qBVz2R4CSZZC0` is auto-converted to `username user1 secret 9 $14$dNmW$QykGZEEGmiEGrE$C9D/fD0czicOtgaZAa1CTa2sgygi0Leyw3/cLqPY426`. After the device is upgraded, run the **write memory** command in privileged EXEC mode for the convoluted type 9 secret to be permanently written into the startup configuration.

- Plain text passwords are converted to nonreversible encrypted password type 9.

**Note** This is supported in Cisco IOS XE Gibraltar 16.10.1 and later releases.

- Secret password type 4 is not supported.

# Information About Controlling Switch Access with Passwords and Privileges

This section provides information about controlling switch access with passwords and privileges.

## Preventing Unauthorized Access

You can prevent unauthorized users from reconfiguring your switch and viewing configuration information. Typically, you want network administrators to have access to your switch while you restrict access to users who dial from outside the network through an asynchronous port, connect from outside the network through a serial port, or connect through a terminal or workstation from within the local network.

To prevent unauthorized access into your switch, you should configure one or more of these security features:

- At a minimum, you should configure passwords and privileges at each switch port. These passwords are locally stored on the switch. When users attempt to access the switch through a port or line, they must enter the password specified for the port or line before they can access the switch.
- For an additional layer of security, you can also configure username and password pairs, which are locally stored on the switch. These pairs are assigned to lines or ports and authenticate each user before that user can access the switch. If you have defined privilege levels, you can also assign a specific privilege level (with associated rights and privileges) to each username and password pair.

- If you want to use username and password pairs, but you want to store them centrally on a server instead of locally, you can store them in a database on a security server. Multiple networking devices can then use the same database to obtain user authentication (and, if necessary, authorization) information.
- You can also enable the login enhancements feature, which logs both failed and unsuccessful login attempts. Login enhancements can also be configured to block future login attempts after a set number of unsuccessful attempts are made.

# Default Password and Privilege Level Configuration

A simple way of providing terminal access control in your network is to use passwords and assign privilege levels. Password protection restricts access to a network or network device. Privilege levels define what commands users can enter after they have logged into a network device.

This table shows the default password and privilege level configuration.

***Table 1: Default Password and Privilege Levels***

| Feature | Default Setting |
|---|---|
| Enable password and privilege level | No password is defined. The default is level 15 (privileged EXEC level). The password is not encrypted in the configuration file. |
| Enable secret password and privilege level | No password is defined. The default is level 15 (privileged EXEC level). The password is encrypted before it is written to the configuration file. |
| Line password | No password is defined. |

# Additional Password Security

To provide an additional layer of security, particularly for passwords that cross the network or that are stored on a Trivial File Transfer Protocol (TFTP) server, you can use either the **enable password** or **enable secret** global configuration commands. Both commands accomplish the same thing; that is, you can establish an encrypted password that users must enter to access privileged EXEC mode (the default) or any privilege level you specify.

We recommend that you use the **enable secret** command because it uses an improved encryption algorithm.

If you configure the **enable secret** command, it takes precedence over the **enable password** command; the two commands cannot be in effect simultaneously.

If you enable password encryption, it applies to all passwords including username passwords, authentication key passwords, the privileged command password, and console and virtual terminal line passwords.

### Unmasked Secret Password

To provide an additional layer of security, particularly for passwords that cross the network or that are stored on a Trivial File Transfer Protocol (TFTP) server, you can use either the **enable password** or **enable secret** global configuration commands. Both commands accomplish the same thing; that is, you can establish an encrypted password that users must enter to access privileged EXEC mode (the default) or any privilege level you specify.

We recommend that you use the **enable secret** command because it uses an improved encryption algorithm.

If you configure the **enable secret** command, it takes precedence over the **enable password** command; the two commands cannot be in effect simultaneously.

If you enable password encryption, it applies to all passwords including username passwords, authentication key passwords, the privileged command password, and console and virtual terminal line passwords.

### Masked Secret Password

With **enable secret** command, password is encrypted but is visible on the terminal when you type the password. To mask the password on the terminal, use the **masked-secret** global configuration command. The encryption type for this password is type 9, by default.

You can use this command to configure masked secret password for common criteria policy.

## Password Recovery

By default, any end user with physical access to the switch can recover from a lost password by interrupting the boot process while the switch is powering on and then by entering a new password.

The password-recovery disable feature protects access to the switch password by disabling part of this functionality. When this feature is enabled, the end user can interrupt the boot process only by agreeing to set the system back to the default configuration. With password recovery disabled, you can still interrupt the boot process and change the password, but the configuration file (config.text) and the VLAN database file (vlan.dat) are deleted.

If you disable password recovery, we recommend that you keep a backup copy of the configuration file on a secure server in case the end user interrupts the boot process and sets the system back to default values. Do not keep a backup copy of the configuration file on the switch. If the switch is operating in VTP transparent mode, we recommend that you also keep a backup copy of the VLAN database file on a secure server. When the switch is returned to the default system configuration, you can download the saved files to the switch by using the Xmodem protocol.

To re-enable password recovery, use the **no system disable password recovery switch** *number* | *all* global configuration command.

## Terminal Line Telnet Configuration

When you power-up your switch for the first time, an automatic setup program runs to assign IP information and to create a default configuration for continued use. The setup program also prompts you to configure your switch for Telnet access through a password. If you did not configure this password during the setup program, you can configure it when you set a Telnet password for a terminal line.

## Username and Password Pairs

You can configure username and password pairs, which are locally stored on the switch. These pairs are assigned to lines or ports and authenticate each user before that user can access the switch. If you have defined privilege levels, you can also assign a specific privilege level (with associated rights and privileges) to each username and password pair.

# Privilege Levels

Cisco devices use privilege levels to provide password security for different levels of switch operation. By default, the Cisco IOS XE software operates in two modes (privilege levels) of password security: user EXEC (Level 1) and privileged EXEC (Level 15). You can configure up to 16 hierarchical levels of commands for each mode. By configuring multiple passwords, you can allow different sets of users to have access to specified commands.

### Privilege Levels on Lines

Users can override the privilege level you set using the **privilege level** line configuration command by logging in to the line and enabling a different privilege level. They can lower the privilege level by using the **disable** command. If users know the password to a higher privilege level, they can use that password to enable the higher privilege level. You might specify a high level or privilege level for your console line to restrict line usage.

For example, if you want many users to have access to the **clear line** command, you can assign it level 2 security and distribute the level 2 password fairly widely. But if you want more restricted access to the **configure** command, you can assign it level 3 security and distribute that password to a more restricted group of users.

### Command Privilege Levels

When you set a command to a privilege level, all commands whose syntax is a subset of that command are also set to that level. For example, if you set the **show ip traffic** command to level 15, the **show** commands and **show ip** commands are automatically set to privilege level 15 unless you set them individually to different levels.

# AES Password Encryption and Master Encryption Keys

You can enable strong, reversible 128-bit Advanced Encryption Standard (AES) password encryption, also known as type 6 encryption. To start using type 6 encryption, enable the AES Password Encryption feature and configure a master encryption key to encrypt and decrypt passwords.

After you enable AES password encryption and configure a master key, all the existing and newly created cleartext passwords for the supported applications are stored in type 6 encrypted format, unless you disable type 6 password encryption. You can also configure the device to convert all the existing weakly encrypted passwords to type 6 encrypted passwords.

Type 0 and 7 passwords can be autoconverted to type 6 if the AES Password Encryption feature and master encryption key are configured.

**Note**

- Type 6 CLI support for TACACS and RADIUS related keys is supported from Cisco IOS XE Everest 16.5.x and later releases.
- Type 6 encrypted password for the username password is supported from Cisco IOS XE Gibraltar 16.10.1 and later releases. Autoconversion to password type 6 is supported from Cisco IOS XE Gibraltar 16.11.1 and later releases.
- Type 6 username and password are backward compatible to Cisco IOS XE Gibraltar 16.10.x. If you downgrade to any release earlier than Cisco IOS XE Gibraltar 16.10.1, the type 6 username and password are rejected. After autoconversion, to prevent an administrator password from getting rejected during a downgrade, migrate the passwords used for administrator logins (management access) to irreversible password types manually.

# How to Configure Switch Access with Passwords and Privileges

## Setting or Changing a Static Enable Password

The enable password controls access to the privileged EXEC mode. Follow these steps to set or change a static enable password:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **enable password** *password*<br>**Example:**<br>`Device(config)# enable password secret321` | Defines a new password or changes an existing password for access to privileged EXEC mode.<br>By default, no password is defined.<br>For *password*, specify a string from 1 to 25 alphanumeric characters. The string cannot start with a number, is case sensitive, and allows spaces but ignores leading spaces. It can contain the question mark (?) character if you precede the question mark with the key combination Crtl-v when you create the password; for |

| | Command or Action | Purpose |
|---|---|---|
| | | example, to create the password abc?123, do this:<br>**a.** Enter **abc**.<br>**b.** Enter **Crtl-v**.<br>**c.** Enter **?123**.<br>When the system prompts you to enter the enable password, you need not precede the question mark with the Ctrl-v; you can simply enter abc?123 at the password prompt. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Protecting Enable and Enable Secret Passwords with Encryption

Follow these steps to establish an encrypted password that users must enter to access privileged EXEC mode (the default) or any privilege level you specify:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode. Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | Use one of the following:<br>• **enable password** [**level** *level*] {*unencrypted-password* \| *encryption-type encrypted-password*}<br>• **enable secret** [**level** *level*] {*unencrypted-password* \| *encryption-type encrypted-password*}<br>**Example:**<br>`Device(config)# enable password level 12 example123` | • Defines a new password or changes an existing password for access to privileged EXEC mode.<br>• Defines a secret password, which is saved using a nonreversible encryption method.<br>  • (Optional) For *level*, the range is from 0 to 15. Level 1 is normal user EXEC mode privileges. The default level is 15 (privileged EXEC mode privileges). |

| | Command or Action | Purpose |
|---|---|---|
| | or<br>`Device(config)# enable secret 9 $9$sMLBsTFXLnnHTk$0L82` | • For *unencrypted-password*, specify a string from 1 to 25 alphanumeric characters. The string cannot start with a number, is case sensitive, and allows spaces but ignores leading spaces. By default, no password is defined.<br>• For *encryption-type*, the available options for **enable password** are type 0 and 7, and type 0, 5, 8, and 9 for **enable secret**. If you specify an encryption type, you must provide an encrypted password—an encrypted password that you copy from another switch configuration. Secret encryption type 9 is more secure, so we recommend that you select type 9 to avoid any issues while upgrading or downgrading.<br>**Note**<br>• If you do not specify an encryption type for the secret password, the password is auto converted to type 9. This is applicable in Cisco IOS XE Gibraltar 16.10.1 and later releases.<br>• If you specify an encryption type and then enter a clear text password, it will result in an error.<br>• You can also configure type 9 encryption for the secret password manually by using the **algorithm-type scrypt** command in global configuration mode. For example:<br>`Device(config)# username user1 algorithm-type scrypt secret cisco`<br>Or<br>`Device(config)# enable algorithm-type scrypt secret cisco` |

| | Command or Action | Purpose |
|---|---|---|
| | | Run the **write memory** command in privileged EXEC mode for the type 9 secret to be permanently written into the startup configuration. |
| **Step 4** | **service password-encryption**<br>**Example:**<br>`Device(config)# service password-encryption` | (Optional) Encrypts the password when the password is defined or when the configuration is written.<br>Encryption prevents the password from being readable in the configuration file. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Disabling Password Recovery

Follow these steps to disable password recovery to protect the security of your switch:

### Before you begin

If you disable password recovery, we recommend that you keep a backup copy of the configuration file on a secure server in case the end user interrupts the boot process and sets the system back to default values. Do not keep a backup copy of the configuration file on the switch. If the switch is operating in VTP transparent mode, we recommend that you also keep a backup copy of the VLAN database file on a secure server. When the switch is returned to the default system configuration, you can download the saved files to the switch by using the Xmodem protocol.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **system disable password recovery switch** {*all* \| *<1-9>*}<br>**Example:**<br>`Device(config)# system disable password recovery switch all` | Disables password recovery.<br>• *all*: Sets the configuration on switches in stack.<br>• *<1-9>*: Sets the configuration on the switch number selected.<br>This setting is saved in an area of the flash memory that is accessible by the boot loader and the Cisco IOS image, but is not a part of the file system and is not accessible by any user. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

### What to do next

To remove **disable password recovery**, use the **no system disable password recovery switch all** global configuration command.

# Setting a Telnet Password for a Terminal Line

Beginning in user EXEC mode, follow these steps to set a Telnet password for the connected terminal line:

### Before you begin

- Attach a PC or workstation with emulation software to the switch console port, or attach a PC to the Ethernet management port.
- The default data characteristics of the console port are 9600, 8, 1, no parity. You might need to press the Return key several times to see the command-line prompt.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **line vty 0 98**<br>**Example:**<br>`Device(config)# line vty 0 98` | Configures the number of Telnet sessions (lines), and enters line configuration mode.<br>There are 99 possible sessions on a command-capable device. The 0 and 98 mean that you are configuring all 99 possible Telnet sessions. |
| **Step 4** | **password** *password*<br>**Example:**<br>`Device(config-line)# password abcxyz543` | Sets a Telnet password for the line or lines.<br>For *password*, specify a string from 1 to 25 alphanumeric characters. The string cannot start with a number, is case sensitive, and allows spaces but ignores leading spaces. By default, no password is defined. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-line)# end` | Returns to privileged EXEC mode. |

# Configuring Username and Password Pairs

Follow these steps to configure username and password pairs:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **username** *name* [**privilege** *level*] {**password** *encryption-type password*}<br>**Example:**<br>`Device(config)# username adamsample privilege 1 password secret456`<br>`Device(config)# username 111111111111 mac attribute` | Sets the username, privilege level, and password for each user.<br>• For *name*, specify the user ID as one word or the MAC address. Spaces and quotation marks are not allowed.<br>• You can configure a maximum of 12000 clients each, for both username and MAC filter. |

| | Command or Action | Purpose |
|---|---|---|
| | | • (Optional) For *level*, specify the privilege level the user has after gaining access. The range is 0 to 15. Level 15 gives privileged EXEC mode access. Level 1 gives user EXEC mode access.<br>• For *encryption-type*, enter **0** to specify that an unencrypted password will follow. Enter **7** to specify that a hidden password will follow. Enter **6** to specify that an encrypted password will follow.<br>• For *password*, specify the password the user must enter to gain access to the device. The password must be from 1 to 25 characters, can contain embedded spaces, and must be the last option specified in the **username** command. |
| **Step 4** | Use one of the following:<br>• **line console 0**<br>• **line vty 0 98**<br>**Example:**<br>`Device(config)# line console 0`<br>or<br>`Device(config)# line vty 0 98` | Enters line configuration mode, and configures the console port (line 0) or the VTY lines (line 0 to 98). |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-line)# end` | Exits line configuration mode and returns to privileged EXEC mode. |

# Setting the Privilege Level for a Command

Follow these steps to set the privilege level for a command:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **privilege** *mode* **level** *level command*<br>**Example:**<br>`Device(config)# privilege exec level 14 configure` | Sets the privilege level for a command.<br>• For *mode*, enter **configure** for global configuration mode, **exec** for EXEC mode, **interface** for interface configuration mode, or **line** for line configuration mode.<br>• For *level*, the range is from 0 to 15. Level 1 is for normal user EXEC mode privileges. Level 15 is the level of access permitted by the **enable** password.<br>• For *command*, specify the command to which you want to restrict access. |
| **Step 4** | **enable password level** *level password*<br>**Example:**<br>`Device(config)# enable password level 14 SecretPswd14` | Specifies the password to enable the privilege level.<br>• For *level*, the range is from 0 to 15. Level 1 is for normal user EXEC mode privileges.<br>• For *password*, specify a string from 1 to 25 alphanumeric characters. The string cannot start with a number, is case sensitive, and allows spaces but ignores leading spaces. By default, no password is defined. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Changing the Default Privilege Level for Lines

Follow these steps to change the default privilege level for the specified line:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** | Enables privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device> enable` | Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **line vty** *line*<br>**Example:**<br>`Device(config)# line vty 10` | Selects the virtual terminal line on which to restrict access. |
| **Step 4** | **privilege exec level** *level*<br>**Example:**<br>`Device(config-line)# privilege exec level 15` | Changes the default privilege level for the line.<br>For *level*, the range is from 0 to 15. Level 1 is for normal user EXEC mode privileges. Level 15 is the level of access permitted by the **enable** password. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-line)# end` | Exits line configuration mode and returns to privileged EXEC mode. |

### What to do next

Users can override the privilege level you set using the **privilege level** line configuration command by logging in to the line and enabling a different privilege level. They can lower the privilege level by using the **disable** command. If users know the password to a higher privilege level, they can use that password to enable the higher privilege level. You might specify a high level or privilege level for your console line to restrict line usage.

# Logging in to and Exiting a Privilege Level

Beginning in user EXEC mode, follow these steps to log into a specified privilege level and exit a specified privilege level.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** *level*<br>**Example:** | Logs in to a specified privilege level.<br>InI the example, Level 15 is privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device> enable 15` | For *level*, the range is 0 to 15. |
| **Step 2** | **disable** *level*<br>**Example:**<br>`Device# disable 1` | Exits to a specified privilege level.<br>In the example, Level 1 is user EXEC mode.<br>For *level*, the range is 0 to 15. |

# Configuring an Encrypted Preshared Key

To configure an encrypted preshared key, perform the following steps.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **key config-key password-encrypt** [*text*]<br>**Example:**<br>`Device(config)# key config-key password-encrypt` | Stores a type 6 encryption key in private NVRAM.<br>• To key in interactively (using the **Enter** key) and an encrypted key already exists, you will be prompted for the following: Old key, New key, and Confirm key.<br>• To key in interactively, but an encryption key is not present, you will be prompted for the following: New key and Confirm key.<br>• When removing the password that is already encrypted, you will see the following prompt:<br>`WARNING: All type 6 encrypted keys will become unusable. Continue with master key deletion? [yes/no]:"` |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **password encryption aes**<br>**Example:**<br>`Device(config)# password encryption aes` | Enables the encrypted preshared key. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Monitoring Switch Access with Passwords and Privileges

*Table 2: Commands for Displaying Privilege-Level Information*

| Command | Information |
|---|---|
| **show privilege** | Displays the privilege level configuration. |

# Configuration Examples for Switch Access with Passwords and Privilege Levels

## Example: Setting or Changing a Static Enable Password

The following example shows how to change the enable password to *l1u2c3k4y5*. The password is not encrypted and provides access to level 15 (traditional privileged EXEC mode access):

```
Device> enable
Device# configure terminal
Device(config)# enable password l1u2c3k4y5
Device(config)# end
```

## Example: Protecting Enable and Enable Secret Passwords with Encryption

The following example shows how to configure the encrypted password *$9$sMLBsTFXLnnHTk$0L82* for privilege level 2:

```
Device> enable
Device# configure terminal
Device(config)# enable secret level 2 9 $9$sMLBsTFXLnnHTk$0L82
Device(config)# end
```

## Example: Setting a Telnet Password for a Terminal Line

The following example shows how to set the Telnet password to *let45me67in89*:

```
Device> enable
Device# configure terminal
Device(config)# line vty 10
Device(config-line)# password let45me67in89
Device(config-line)# end
```

## Example: Setting the Privilege Level for a Command

The following example shows how to set the **configure** command to privilege level 14 and define *SecretPswd14* as the password users must enter to use level 14 commands:

```
Device> enable
Device# configure terminal
Device(config)# privilege exec level 14 configure
Device(config)# enable password level 14 SecretPswd14
Device(config)# end
```

## Example: Configuring an Encrypted Preshared Key

The following example shows a configuration for which a type 6 preshared key has been encrypted. It includes the prompts and messages that a user might see.

```
Device> enable
Device# configure terminal
Device(config)# password encryption aes
Device(config)# key config-key password-encrypt
New key:
Confirm key:
Device(config)#
01:46:40: TYPE6_PASS: New Master key configured, encrypting the keys with
the new master key
Device(config)# end
```

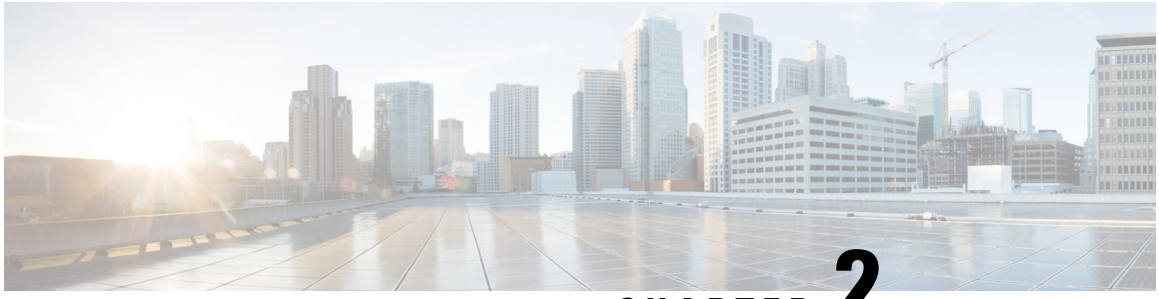
# Feature History for Controlling Switch Access with Passwords and Privileges

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Controlling Switch Access with Passwords and Privileges | Password protection restricts access to a network or network device. Privilege levels define what commands users can enter after they have logged into a network device.<br><br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Controlling Switch Access with Passwords and Privileges | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Gibraltar 16.11.1 | Autoconversion of Type 0 and Type 7 Username and Password to Type 6 | From this release, type 0 and 7 username and password can be autoconverted to type 6.<br><br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches.<br><br>Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.



CHAPTER 2

# Configuring Login Block


- Information About Login Enhancements-Login Block, on page 19
- How to Configure Login Enhancements-Login Block, on page 20
- Verifying Login Parameters, on page 22
- Configuration Examples for Login Enhancements-Login Block, on page 24
- Feature History for Login Enhancements-Login Block, on page 24


# Information About Login Enhancements-Login Block

## Login Enhancements-Login Block Overview

The Login Enhancements (Login Block) feature allows users to enhance the security of a device by configuring options to automatically block further login attempts when a possible denial-of-service (DoS) attack is detected.

The login block and login delay options introduced by this feature can be configured for Telnet or SSH virtual connections. By enabling this feature, you can slow down “dictionary attacks” by enforcing a “quiet period” if multiple failed connection attempts are detected, thereby protecting the routing device from a type of denial-of-service attack.

## Protecting Against Denial of Service and Dictionary Login Attacks

Connecting to a device for the purposes of administering (managing) the device, at either the User or Executive level, is most frequently performed using Telnet or SSH (secure shell) from a remote console (such as a PC). SSH provides a more secure connection option because communication traffic between the user’s device and the managed device are encrypted. The Login Block capability, when enabled, applies to both Telnet connections and SSH connections. Beginning in Release versions 12.3(33)SRB2, 12.2(33)SXH2, and 12.4(15)T1, the Login Block capability also applies to HTTP connections.”

The automated activation and logging of the Login Block and Quiet Period capabilities introduced by this feature are designed to further enhance the security of your devices by specifically addressing two well known methods that individuals use to attempt to disrupt or compromise network devices.

If the connection address of a device is discovered and is reachable, a malicious user may attempt to interfere with the normal operations of the device by flooding it with connection requests. This type of attack is referred to as an attempted Denial-of-Service, because it is possible that the device may become too busy trying to process the repeated login connection attempts to properly handle normal routing services or are not able to provide the normal login service to legitimate system administrators.

The primary intention of a dictionary attack, unlike a typical DoS attack, is to actually gain administrative access to the device. A dictionary attack is an automated process to attempt to login by attempting thousands, or even millions, of username/password combinations. (This type of attack is called a “dictionary attack” because it typically uses, as a start, every word found in a typical dictionary as a possible password.) As scripts or programs are used to attempt this access, the profile for such attempts is typically the same as for DoS attempts; multiple login attempts in a short period of time.

By enabling a detection profile, the device can be configured to react to repeated failed login attempts by refusing further connection request (login blocking). This block can be configured for a period of time, called a “quiet period”. Legitimate connection attempts can still be permitted during a quiet period by configuring an access-list (ACL) with the addresses that you know to be associated with system administrators.

## Delays Between Successive Login Attempts

A device can accept virtual connections as fast as they can be processed. Introducing a delay between login attempts helps to protect the device against malicious login connections such as dictionary attacks and DoS attacks. Delays can be enabled in one of the following ways:

- Through the **auto secure** command. If you enable the AutoSecure feature, the default login delay time of one second is automatically enforced.
- Through the **login block-for** command. You must enter this command before issuing the **login delay** command. If you enter only the **login block-for** command, the default login delay time of one second is automatically enforced.
- Through the global configuration mode command, **login delay**, which allows you to specify login delay time to be enforced, in seconds.

## Login Shutdown If DoS Attacks Are Suspected

If the configured number of connection attempts fail within a specified time period, the device does not accept any additional connections for a “quiet period.” (Hosts that are permitted by a predefined access-control list [ACL] are excluded from the quiet period.)

The number of failed connection attempts that trigger the quiet period can be specified through the new global configuration mode command **login block-for**. The predefined ACL that is excluded from the quiet period can be specified through the new global configuration mode command **login quiet-mode access-class**.

This functionality is disabled by default, and it is not enabled if AutoSecure if enabled.

# How to Configure Login Enhancements-Login Block

## Configuring Login Parameters

Use this task to configure your device for login parameters that help detect suspected DoS attacks and slow down dictionary attacks.

All login parameters are disabled by default. You must issue the **login block-for** command, which enables default login functionality, before using any other login commands. After the **login block-for** command is enabled, the following defaults are enforced:

- A default login delay of one second
- All login attempts made through Telnet or SSH are denied during the quiet period; that is, no ACLs are exempt from the login period until the **login quiet-mode access-class** command is issued.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **login block-for** *seconds* **attempts** *tries* **within** *seconds*<br>**Example:**<br>`Device(config)# login block-for 100 attempts 2 within 100` | Configures your device for login parameters that help provide DoS detection.<br>**Note**<br>This command must be issued before any other login command can be used. |
| **Step 4** | **login quiet-mode access-class** {*acl-name* \| *acl-number*}<br>**Example:**<br>`Device(config)# login quiet-mode access-class myacl` | (Optional) Although this command is optional, it is recommended that it be configured to specify an ACL that is to be applied to the device when the device switches to quiet mode. When the device is in quiet mode, all login requests are denied and the only available connection is through the console.<br>If this command is not configured, then the default ACL **sl_def_acl** is created on the device. This ACL is hidden in the running configuration. Use the **show access-list sl_def_acl** to view the parameters for the default ACL.<br>For example:<br>`Device# show access-lists sl_def_acl`<br>`Extended IP access list sl_def_acl`<br>`10 deny tcp any any eq telnet`<br>`20 deny tcp any any eq www`<br>`30 deny tcp any any eq 22`<br>`40 permit ip any any` |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **login delay** *seconds*<br>**Example:**<br>`Device(config)# login delay 10` | (Optional) Configures a delay between successive login attempts. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and returns to privileged EXEC mode. |
| **Step 7** | **show login failures**<br>**Example:**<br>`Device# show login` | Displays login parameters.<br>• **failures** --Displays information related only to failed login attempts. |

# Verifying Login Parameters

Use this task to verify the applied login configuration and present login status on your device.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **show login failures**<br>**Example:**<br>`Device# show login` | Displays login parameters.<br>• **failures** --Displays information related only to failed login attempts. |

### Examples

The following sample output from the **show login** command verifies that no login parameters have been specified:

```
Device# show login

No login delay has been applied.
No Quiet-Mode access list has been configured.
All successful login is logged and generate SNMP traps.
All failed login is logged and generate SNMP traps
Router NOT enabled to watch for login Attacks
```

The following sample output from the **show login** command verifies that the **login block-for**command is issued. In this example, the command is configured to block login hosts for 100 seconds if 16 or more login requests fail within 100 seconds; five login requests have already failed.

```
Device# show login

A default login delay of 1 seconds is applied.
No Quiet-Mode access list has been configured.
All successful login is logged and generate SNMP traps.
All failed login is logged and generate SNMP traps.
Router enabled to watch for login Attacks.
If more than 15 login failures occur in 100 seconds or less, logins will be disabled for
100 seconds.
Router presently in Watch-Mode, will remain in Watch-Mode for 95 seconds.
Present login failure count 5.
```

The following sample output from the **show login** command verifies that the device is in quiet mode. In this example, the **login block-for** command was configured to block login hosts for 100 seconds if 3 or more login requests fail within 100 seconds.

```
Device# show login

A default login delay of 1 seconds is applied.
No Quiet-Mode access list has been configured.
All successful login is logged and generate SNMP traps.
All failed login is logged and generate SNMP traps.
Router enabled to watch for login Attacks.
If more than 2 login failures occur in 100 seconds or less, logins will be disabled for 100
 seconds.
Router presently in Quiet-Mode, will remain in Quiet-Mode for 93 seconds.
Denying logins from all sources.
```

The following sample output from **show login failures** command shows all failed login attempts on the device:

```
Device# show login failures

Information about login failure's with the device
Username      Source IPAddr  lPort Count  TimeStamp
try1          10.1.1.1        23    1     21:52:49 UTC Sun Feb 24 2019
try2          10.1.1.2        23    1     21:52:52 UTC Sun Feb 23 2019
```

The following sample output from **show login failures** command verifies that no information is presently logged:

```
Device# show login failures

*** No logged failed login attempts with the device.***
```

# Configuration Examples for Login Enhancements-Login Block

## Example: Configuring Login Parameters

The following example shows how to configure your device to enter a 100 second quiet period if 15 failed login attempts is exceeded within 100 seconds; all login requests are denied during the quiet period except hosts from the ACL "myacl."

```
Device> enable
Device# configure terminal
Device(config)# login block-for 100 attempts 15 within 100
Device(config)# login quiet-mode access-class myacl
Device(config)# end
```

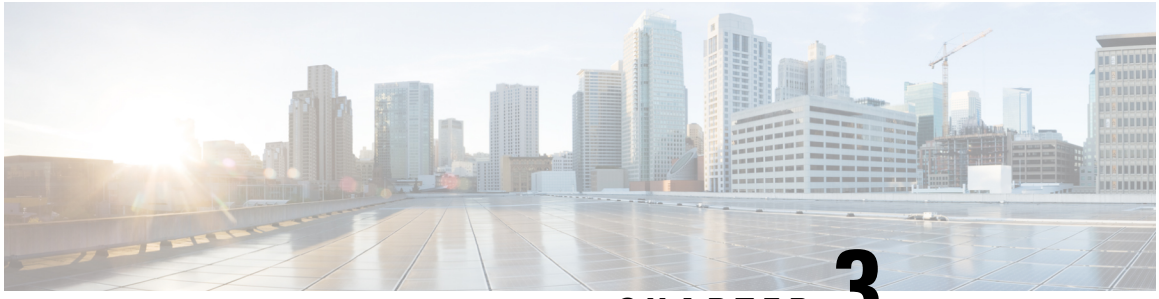
# Feature History for Login Enhancements-Login Block

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Login Enhancements -Login Block | The Login Enhancements-Login Block feature allows users to enhance the security of a router by configuring options to automatically block further login attempts when a possible DoS attack is detected.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Login Enhancements -Login Block | Support for this feature was introduced on the C9500-24Y4C, C9500-32C, C9500-32QC, and C9500-48Y4C models of Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.



CHAPTER 3

# Configuring Authentication

Authentication provides a method to identify users, which includes the login and password dialog, challenge and response, messaging support, and encryption, depending on the selected security protocol. Authentication is the way a user is identified prior to being allowed access to the network and network services.


- Prerequisites for Configuring Authentication, on page 25
- Restrictions for Configuring Authentication, on page 25
- Information About Authentication, on page 25
- How to Configure Authentication, on page 41
- Configuration Examples for Authentication, on page 61
- Feature History for Configuring Authentication, on page 74


# Prerequisites for Configuring Authentication

The implementation of authentication is divided into Authentication, Authorization, and Accounting (AAA) authentication and nonauthentication methods. Cisco recommends that, whenever possible, AAA security services be used to implement authentication.

# Restrictions for Configuring Authentication

- The number of AAA method lists that can be configured is 250.
- If you configure the same RADIUS server IP address for a different UDP destination port for accounting requests by using the **acct-port** keyword and a UDP destination port for authentication requests by using the **auth-port** keyword with and without the nonstandard option, the RADIUS server does not accept the nonstandard option.

# Information About Authentication

## Named Method Lists for Authentication

A named list of authentication methods is first defined before AAA authentication can be configured, and the named list is then applied to various interfaces. The method list defines the types of authentication and the

sequence in which they are performed; it must be applied to a specific interface before any of the defined authentication methods are performed. The only exception is the default method list (which is named “default”). The default method list is automatically applied to all interfaces, except those that have a named method list explicitly defined. A defined method list overrides the default method list.

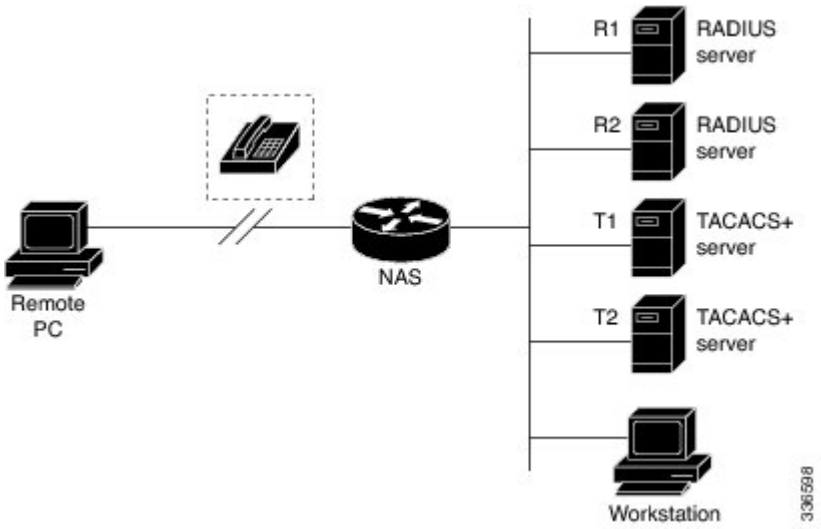
A method list is a sequential list describing the authentication methods to be queried to authenticate a user. Method lists enable you to designate one or more security protocols to be used for authentication, thus ensuring a backup system for authentication in case the initial method fails. Cisco software uses the first listed method to authenticate users. If that method fails to respond, the Cisco software selects the next authentication method listed in the method list. This process continues until there is successful communication with a listed authentication method, or all methods defined in the method list are exhausted.

Note that the software attempts authentication with the next listed authentication method only when there is no response from the previous method. If authentication fails at any point in this cycle, that is, the security server or local username database responds by denying the user access, then the authentication process stops and no other authentication methods are attempted.

## Method Lists and Server Groups

A server group is a way to group existing RADIUS or TACACS+ server hosts for use in method lists. The figure below shows a typical AAA network configuration that includes four security servers: R1 and R2 are RADIUS servers and T1 and T2 are TACACS+ servers. R1 and R2 make up the group of RADIUS servers. T1 and T2 make up the group of TACACS+ servers.

***Figure 1: Typical AAA Network Configuration***




Using server groups, you can specify a subset of the configured server hosts and use them for a particular service. For example, server groups allow you to define R1 and R2 as a server group, and define T1 and T2 as a separate server group. For example, you can specify R1 and T1 in the method list for authentication login, while specifying R2 and T2 in the method list for PPP authentication.

Server groups also can include multiple host entries for the same server, as long as each entry has a unique identifier. The combination of an IP address and a UDP port number creates a unique identifier, allowing different ports to be individually defined as RADIUS hosts providing a specific AAA service. In other words, this unique identifier enables RADIUS requests to be sent to different UDP ports on a server at the same IP address. If two different host entries on the same RADIUS server are configured for the same service--for

example, authentication--the second host entry configured acts as failover backup to the first one. Using this example, if the first host entry fails to provide accounting services, the network access server will try the second host entry configured on the same device for accounting services. (The RADIUS host entries will be tried in the order in which they are configured.)

For more information about configuring server groups and about configuring server groups based on Dialed Number Identification Service (DNIS) numbers, see the "Configuring RADIUS" or "Configuring TACACS+" chapters.

# Login Authentication Using AAA

## Login Authentication Using Enable Password

Use the **aaa authentication login**command with the **enable** keyword to specify the enable password as the login authentication method. For example, to specify the enable password as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default enable
```

Before you can use the enable password as the login authentication method, you need to define the enable password. For more information about defining enable passwords, see chapter "Controlling Switch Access with Passwords and Privilege Levels."

## Login Authentication Using Kerberos

Authentication via Kerberos is different from most other authentication methods: the user's password is never sent to the remote access server. Remote users logging in to the network are prompted for a username. If the key distribution center (KDC) has an entry for that user, it creates an encrypted ticket granting ticket (TGT) with the password for that user and sends it back to the device. The user is then prompted for a password, and the device attempts to decrypt the TGT with that password. If it succeeds, the user is authenticated and the TGT is stored in the user's credential cache on the device.

While krb5 does use the KINIT program, a user does not need to run the KINIT program to get a TGT to authenticate to the device. This is because KINIT has been integrated into the login procedure in the Cisco IOS XE implementation of Kerberos.

Use the **aaa authentication login**command with the **krb5** keyword to specify Kerberos as the login authentication method. For example, to specify Kerberos as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default krb5
```

Before you can use Kerberos as the login authentication method, you need to enable communication with the Kerberos security server. For more information about establishing communication with a Kerberos server, refer to the chapter "Configuring Kerberos."

## Login Authentication Using Line Password

Use the **aaa authentication login default** command with the **line** keyword to specify the line password as the login authentication method. For example, to specify the line password as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default line
```

Before you can use a line password as the login authentication method, you need to define a line password.

## Login Authentication Using Local Password

Use the **aaa authentication login default**command with the **local** keyword to specify that the Cisco device will use the local username database for authentication. For example, to specify the local username database as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default local
```

## Login Authentication Using Group RADIUS

Use the **aaa authentication login default**command with the **group radius** to specify RADIUS as the login authentication method. For example, to specify RADIUS as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default group radius
```

Before you can use RADIUS as the login authentication method, you need to enable communication with the RADIUS security server. For more information about establishing communication with a RADIUS server, refer to the chapter “Configuring RADIUS.”

### RADIUS Attribute 8 in Access Requests

After you have used the **aaa authentication login** command to specify RADIUS and your login host has been configured to request its IP address from the NAS, you can send attribute 8 (Framed-IP-Address) in access-request packets by using the **radius-server attribute 8 include-in-access-req** command in global configuration mode. This command makes it possible for NAS to provide the RADIUS server a hint of the user IP address in advance for user authentication.

## Login Authentication Using Group TACACS

Use the **aaa authentication login default** command with the **group tacacs+** to specify TACACS+ as the login authentication method. For example, to specify TACACS+ as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default group tacacs+
```

Before you can use TACACS+ as the login authentication method, you need to enable communication with the TACACS+ security server. For more information about establishing communication with a TACACS+ server, refer to the chapter “Configuring TACACS+.”

**Note** If the **ip radius source interface** command is configured under the group, the MTU of the source interface is used for injecting the packet. And if not configured, default fragmentation at 1396 bytes is done.

## Login Authentication Using Group Name

Use the **aaa authentication login default** command with the **group** *group-name* method to specify a subset of RADIUS or TACACS+ servers to use as the login authentication method. To specify and define the group name and the members of the group, use the **aaa group server** command. For example, use the **aaa group server** command to first define the members of **group loginrad**:

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius loginrad
Device(config-sg-radius)# server 172.16.2.3
Device(config-sg-radius)# server 172.16.2 17
Device(config-sg-radius)# server 172.16.2.32
Device(config-sg-radius)# end
```

This command specifies RADIUS servers 172.16.2.3, 172.16.2.17, and 172.16.2.32 as members of the group *loginrad*.

To specify **group loginrad** as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication login default group loginrad
```

Before you can use a group name as the login authentication method, you need to enable communication with the RADIUS or TACACS+ security server. For more information about establishing communication with a RADIUS server, refer to the chapter “Configuring RADIUS.” For more information about establishing communication with a TACACS+ server, refer to the chapter “Configuring TACACS+.”

# PPP Authentication Using AAA

## PPP Authentication Using Kerberos

Use the **aaa authentication ppp default Device**command with the **krb5** keyword to specify Kerberos as the authentication method for use on interfaces running PPP. For example, to specify Kerberos as the method of user authentication when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication ppp default krb5
```

Before you can use Kerberos as the PPP authentication method, you need to enable communication with the Kerberos security server. For more information about establishing communication with a Kerberos server, refer to the chapter “Configuring Kerberos”.

**Note** Kerberos login authentication works only with PPP PAP authentication.

## PPP Authentication Using Local Password

Use the **aaa authentication ppp default** command with the **local** keyword to specify that the Cisco device will use the local username database for authentication. For example, to specify the local username database as the method of authentication for use on lines running PPP when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication ppp default local
```

## PPP Authentication Using Group RADIUS

Use the **aaa authentication ppp default group radius** command to specify RADIUS as the login authentication method. For example, to specify RADIUS as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication ppp default group radius
```

Before you can use RADIUS as the PPP authentication method, you need to enable communication with the RADIUS security server. For more information about establishing communication with a RADIUS server, refer to the chapter "Configuring RADIUS."

### RADIUS Attribute 44 in Access Requests

After you have used the **aaa authentication ppp default group radius** command to specify RADIUS as the login authentication method, you can configure your device to send attribute 44 (Acct-Session-ID) in access-request packets by using the **radius-server attribute 44 include-in-access-req** command in global configuration mode. This command allows the RADIUS daemon to track a call from the beginning to the end.

## PPP Authentication Using Group TACACS

Use the **aaa authentication ppp default group tacacs+** command to specify TACACS+ as the login authentication method. For example, to specify TACACS+ as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication ppp default group tacacs+
```

Before you can use TACACS+ as the PPP authentication method, you need to enable communication with the TACACS+ security server. For more information about establishing communication with a TACACS+ server, refer to the chapter "Configuring TACACS+."

## PPP Authentication Using Group Name

Use the **aaa authentication ppp default** command with the **group** *group-name* method to specify a subset of RADIUS or TACACS+ servers to use as the login authentication method. To specify and define the group name and the members of the group, use the **aaa group server** command. For example, use the **aaa group server** command to first define the members of **group ppprad**:

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius ppprad
Device(config-sg-radius)# server 172.16.2.3
Device(config-sg-radius)# server 172.16.2 17
Device(config-sg-radius)# server 172.16.2.32
Device(config-sg-radius)# end
```

This command specifies RADIUS servers 172.16.2.3, 172.16.2.17, and 172.16.2.32 as members of the group *ppprad*.

To specify **group ppprad** as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication ppp default group ppprad
```

Before you can use a group name as the PPP authentication method, you need to enable communication with the RADIUS or TACACS+ security server. For more information about establishing communication with a RADIUS server, refer to the chapter "Configuring RADIUS". For more information about establishing communication with a TACACS+ server, refer to the chapter "Configuring TACACS+."

## AAA Scalability for PPP Requests

You can configure and monitor the number of background processes allocated by the PPP manager in the network access server (NAS) to deal with AAA authentication and authorization requests. The AAA Scalability feature enables you to configure the number of processes used to handle AAA requests for PPP, thus increasing the number of users that can be simultaneously authenticated or authorized.

To allocate a specific number of background processes to handle AAA requests for PPP, use the following command in global configuration mode:

```
Device(config)# aaa processes 5000
```

The argument *number* defines the number of background processes earmarked to process AAA authentication and authorization requests for PPP and can be configured for any value from 1 to 2147483647. Because of the way the PPP manager handles requests for PPP, this argument also defines the number of new users that can be simultaneously authenticated. This argument can be increased or decreased at any time.

**Note** Allocating additional background processes can be expensive. You should configure the minimum number of background processes capable of handling the AAA requests for PPP.

# ARAP Authentication Using AAA

## ARAP Authentication Allowing Authorized Guest Logins

Use the **aaa authentication arap default** command with the **auth-guest** keyword to allow guest logins only if the user has already successfully logged in to the EXEC. This method must be the first listed in the ARAP authentication method list but it can be followed by other methods if it does not succeed. For example, to allow all authorized guest logins--meaning logins by users who have already successfully logged in to the EXEC--as the default method of authentication, using RADIUS only if that method fails, enter the following command:

```
Device(config)# aaa authentication arap default auth-guest group radius
```

**Note** By default, guest logins through ARAP are disabled when you initialize AAA. To allow guest logins, you must use the **aaa authentication arap** {*authentication-list* | **default** command with either the **guest** or the **auth-guest** keyword.

## ARAP Authentication Allowing Guest Logins

Use the **aaa authentication arap** {**default** | *authentication-list*} command with the **guest** keyword to allow guest logins. This method must be the first listed in the ARAP authentication method list but it can be followed

by other methods if it does not succeed. For example, to allow all guest logins as the default method of authentication, using RADIUS only if that method fails, enter the following command:

```
Device(config)# aaa authentication arap default guest group radius
```

## ARAP Authentication Using Line Password

Use the **aaa authentication arap** {**default** | *authentication-list*}command with the **line** keyword to specify the line password as the authentication method. For example, to specify the line password as the method of ARAP user authentication when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication arap default line
```

Before you can use a line password as the ARAP authentication method, you need to define a line password.

## ARAP Authentication Using Local Password

Use the **aaa authentication arap** {**default** | *authentication-list*}command with the **local** keyword to specify that the Cisco device will use the local username database for authentication. For example, to specify the local username database as the method of ARAP user authentication when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication arap default local
```

## ARAP Authentication Using Group RADIUS

Use the **aaa authentication arap** {**default** | *authentication-list*}command with the **group radius** *method* to specify RADIUS as the ARAP authentication method. For example, to specify RADIUS as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication arap default group radius
```

Before you can use RADIUS as the ARAP authentication method, you need to enable communication with the RADIUS security server. For more information about establishing communication with a RADIUS server, refer to the chapter “Configuring RADIUS.”

## ARAP Authentication Using Group TACACS

Use the **aaa authentication arap** {**default** | *authentication-list*} command with the **group tacacs+** *method* to specify TACACS+ as the ARAP authentication method. For example, to specify TACACS+ as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication arap default group tacacs+
```

Before you can use TACACS+ as the ARAP authentication method, you need to enable communication with the TACACS+ security server. For more information about establishing communication with a TACACS+ server, refer to the chapter “Configuring TACACS+.”

## ARAP Authentication Using a Group Name

Use the **aaa authentication arap** {**default** | *authentication-list*} command with the **group** *group-name* method to specify a subset of RADIUS or TACACS+ servers to use as the ARAP authentication method. To specify

and define the group name and the members of the group, use the **aaa group server** command. For example, use the **aaa group server**command to first define the members of **group araprad**:

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius araprad
Device(config-sg-radius)# server 172.16.2.3
Device(config-sg-radius)# server 172.16.2 17
Device(config-sg-radius)# server 172.16.2.32
Device(config-sg-radius)# end
```

This command specifies RADIUS servers 172.16.2.3, 172.16.2.17, and 172.16.2.32 as members of the group *araprad*.

To specify **group araprad** as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication arap default group araprad
```

Before you can use a group name as the ARAP authentication method, you need to enable communication with the RADIUS or TACACS+ security server. For more information about establishing communication with a RADIUS server, refer to the chapter “Configuring RADIUS.” For more information about establishing communication with a TACACS+ server, refer to the chapter “Configuring TACACS+.”

# NASI Authentication Using AAA

## NASI Authentication Using Enable Password

Use the **aaa authentication nasi** command with the keyword **enable** to specify the enable password as the authentication method. For example, to specify the enable password as the method of NASI user authentication when no other method list has been defined, use the following command:

```
Devie(config)# aaa authentication nasi default enable
```

Before you can use the enable password as the authentication method, you need to define the enable password.

## NASI Authentication Using Group RADIUS

Use the **aaa authentication nasi** command with the **group radius** method to specify RADIUS as the NASI authentication method. For example, to specify RADIUS as the method of NASI user authentication when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication nasi default group radius
```

Before you can use RADIUS as the NASI authentication method, you need to enable communication with the RADIUS security server.

## NASI Authentication Using Group TACACS

Use the **aaa authentication nasi** command with the **group tacacs+** keyword to specify TACACS+ as the NASI authentication method. For example, to specify TACACS+ as the method of NASI user authentication when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication nasi default group tacacs+
```

Before you can use TACACS+ as the authentication method, you need to enable communication with the TACACS+ security server.

## NASI Authentication Using Line Password

Use the **aaa authentication nasi** command with the keyword **line** to specify the line password as the authentication method. For example, to specify the line password as the method of NASI user authentication when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication nasi default line
```

Before you can use a line password as the NASI authentication method, you need to define a line password.

## NASI Authentication Using Local Password

Use the **aaa authentication nasi** command with the keyword **local** to specify that the Cisco rdevice will use the local username database for authentication information. For example, to specify the local username database as the method of NASI user authentication when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication nasi default local
```

## NASI Authentication Using Group Name

Use the **aaa authentication nasi** command with the **group** *group-name* method to specify a subset of RADIUS or TACACS+ servers to use as the NASI authentication method. To specify and define the group name and the members of the group, use the **aaa group server** command. For example, use the **aaa group server**command to first define the members of **group nasirad**:

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius nasirad
Device(config-sg-radius)# server 172.16.2.3
Device(config-sg-radius)# server 172.16.2 17
Device(config-sg-radius)# server 172.16.2.32
Device(config-sg-radius)# end
```

This command specifies RADIUS servers 172.16.2.3, 172.16.2.17, and 172.16.2.32 as members of the group *nasirad*.

To specify **group nasirad** as the method of user authentication at login when no other method list has been defined, enter the following command:

```
Device(config)# aaa authentication nasi default group nasirad
```

Before you can use a group name as the NASI authentication method, you need to enable communication with the RADIUS or TACACS+ security server.

# Specifying the Amount of Time for Login Input

The **timeout login response**command allows you to specify how long the system will wait for login input (such as username and password) before timing out. The default login value is 30 seconds; with the **timeout**

**login response** command, you can specify a timeout value from 1 to 300 seconds. To change the login timeout value from the default of 30 seconds, use the following command in line configuration mode:

```
Device(config-line)# timeout login response 30
```

# Password Protection at the Privileged Level

Use the **aaa authentication enable default** command to create a series of authentication methods that are used to determine whether a user can access the privileged EXEC command level. You can specify up to four authentication methods. The additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

Use the following command in global configuration mode:

```
Device(config)# authentication enable default radius
```

or

```
Device(config)# authentication enable default tacacs
```

# Changing the Text Displayed at the Password Prompt

Use the **aaa authentication password-prompt** command to change the default text that the Cisco IOS XE software displays when prompting a user to enter a password. This command changes the password prompt for the enable password as well as for login passwords that are not supplied by remote security servers. The **no** form of this command returns the password prompt to the following default value:

```
Password:
```

The **aaa authentication password-prompt** command does not change any dialog that is supplied by a remote TACACS+ or RADIUS server.

The **aaa authentication password-prompt** command works when RADIUS is used as the login method. You will be able to see the password prompt defined in the command shown even when the RADIUS server is unreachable. The **aaa authentication password-prompt** command does not work with TACACS+. TACACS+ supplies the NAS with the password prompt to display to the users. If the TACACS+ server is reachable, the NAS gets the password prompt from the server and uses that prompt instead of the one defined in the **aaa authentication password-prompt** command. If the TACACS+ server is not reachable, the password prompt defined in the **aaa authentication password-prompt** command may be used.

Use the following command in global configuration mode:

```
Device(config)# aaa authentication password-prompt "Enter your password now:"
```

# Double Authentication of PPP Sessions

PPP sessions can be authenticated only by using a single authentication method: either PAP or CHAP. Double authentication requires remote users to pass a second stage of authentication (after CHAP or PAP authentication) before gaining network access.

This second ("double") authentication requires a password that is known to the user but *not* stored on the user's remote host. Therefore, the second authentication is specific to a user, not to a host. This provides an additional level of security that will be effective even if information from the remote host is stolen. In addition, this also provides greater flexibility by allowing customized network privileges for each user.

The second stage authentication can use one-time passwords such as token card passwords, which are not supported by CHAP. If one-time passwords are used, a stolen user password is of no use to the perpetrator.

## How Double Authentication Works

With double authentication, there are two authentication/authorization stages. These two stages occur after a remote user dials in and a PPP session is initiated.

In the first stage, the user logs in using the remote host name; CHAP (or PAP) authenticates the remote host, and then PPP negotiates with AAA to authorize the remote host. In this process, the network access privileges associated with the remote host are assigned to the user.

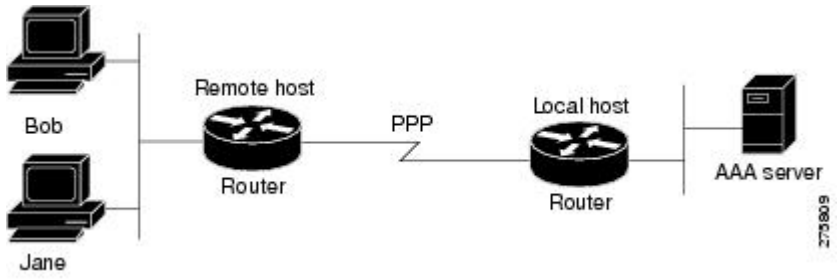
**Note** We suggest that the network administrator restrict authorization at this first stage to allow only Telnet connections to the local host.

In the second stage, the remote user must Telnet to the network access server to be authenticated. When the remote user logs in, the user must be authenticated with AAA login authentication. The user then must enter the **access-profile** command to be reauthorized using AAA. When this authorization is complete, the user has been double authenticated, and can access the network according to per-user network privileges.

The system administrator determines what network privileges remote users will have after each stage of authentication by configuring appropriate parameters on a security server. To use double authentication, the user must activate it by issuing the **access-profile**command.

**Caution** Double authentication can cause certain undesirable events if multiple hosts share a PPP connection to a network access server, as shown in the figure below. First, if a user, Bob, initiates a PPP session and activates double authentication at the network access server (per the figure below), any other user will automatically have the same network privileges as Bob until Bob's PPP session expires. This happens because Bob's authorization profile is applied to the network access server's interface during the PPP session and any PPP traffic from other users will use the PPP session Bob established. Second, if Bob initiates a PPP session and activates double authentication, and then--before Bob's PPP session has expired--another user, Jane, executes the **access-profile** command (or, if Jane Telnets to the network access server and **autocommand access-profile**is executed), a reauthorization will occur and Jane's authorization profile will be applied to the interface--replacing Bob's profile. This can disrupt or halt Bob's PPP traffic, or grant Bob additional authorization privileges Bob should not have.

*Figure 2: Possibly Risky Topology: Multiple Hosts Share a PPP Connection to a Network Access Server*



## Accessing the User Profile After Double Authentication

In double authentication, when a remote user establishes a PPP link to the local host using the local host name, the remote host is CHAP (or PAP) authenticated. After CHAP (or PAP) authentication, PPP negotiates with AAA to assign network access privileges associated with the remote host to the user. (We suggest that privileges at this stage be restricted to allow the user to connect to the local host only by establishing a Telnet connection.)

When the user needs to initiate the second phase of double authentication, establishing a Telnet connection to the local host, the user enters a personal username and password (different from the CHAP or PAP username and password). This action causes AAA reauthentication to occur according to the personal username/password. The initial rights associated with the local host, though, are still in place. By using the **access-profile** command, the rights associated with the local host are replaced by or merged with those defined for the user in the user's profile.

To access the user profile after double authentication, use the following command in EXEC configuration mode:

```
Device> access-profile merge ignore-sanity-checks
```

If you configured the **access-profile** command to be executed as an autocommand, it will be executed automatically after the remote user logs in.

# CHAP or PAP Authentication

One of the most common transport protocols used in ISPs dial solutions is the Point-to-Point Protocol (PPP). Traditionally, remote users dial in to an access server to initiate a PPP session. After PPP has been negotiated, remote users are connected to the ISP network and to the Internet.

Because ISPs want only customers to connect to their access servers, remote users are required to authenticate to the access server before they can start up a PPP session. Normally, a remote user authenticates by typing in a username and password when prompted by the access server. Although this is a workable solution, it is difficult to administer and awkward for the remote user.

A better solution is to use the authentication protocols built into PPP. In this case, the remote user dials in to the access server and starts up a minimal subset of PPP with the access server. This does not give the remote user access to the ISP's network--it merely allows the access server to talk to the remote device.

PPP currently supports two authentication protocols: Password Authentication Protocol (PAP) and Challenge Handshake Authentication Protocol (CHAP). Both are specified in RFC 1334 and are supported on synchronous and asynchronous interfaces. Authentication via PAP or CHAP is equivalent to typing in a username and password when prompted by the server. CHAP is considered to be more secure because the remote user's password is never sent across the connection.

PPP (with or without PAP or CHAP authentication) is also supported in dialout solutions. An access server utilizes a dialout feature when it initiates a call to a remote device and attempts to start up a transport protocol such as PPP.

**Note** To use CHAP or PAP, you must be running PPP encapsulation.

When CHAP is enabled on an interface and a remote device attempts to connect to it, the access server sends a CHAP packet to the remote device. The CHAP packet requests or "challenges" the remote device to respond. The challenge packet consists of an ID, a random number, and the host name of the local device.

When the remote device receives the challenge packet, it concatenates the ID, the remote device's password, and the random number, and then encrypts all of it using the remote device's password. The remote device sends the results back to the access server, along with the name associated with the password used in the encryption process.

When the access server receives the response, it uses the name it received to retrieve a password stored in its user database. The retrieved password should be the same password the remote device used in its encryption process. The access server then encrypts the concatenated information with the newly retrieved password--if the result matches the result sent in the response packet, authentication succeeds.

The benefit of using CHAP authentication is that the remote device's password is never transmitted in clear text. This prevents other devices from stealing it and gaining illegal access to the ISP's network.

CHAP transactions occur only at the time a link is established. The access server does not request a password during the rest of the call. (The local device can, however, respond to such requests from other devices during a call.)

When PAP is enabled, the remote device attempting to connect to the access server is required to send an authentication request. If the username and password specified in the authentication request are accepted, the Cisco IOS XE software sends an authentication acknowledgment.

After you have enabled CHAP or PAP, the access server will require authentication from remote devices dialing in to the access server. If the remote device does not support the enabled protocol, the call will be dropped.

To use CHAP or PAP, you must perform the following tasks:

- Enable PPP encapsulation.
- Enable CHAP or PAP on the interface.
- For CHAP, configure host name authentication and the secret or password for each remote system with which authentication is required.

## Enabling PPP Encapsulation

To enable PPP encapsulation, use the following command in interface configuration mode:

```
Device(config-if)# encapsulation ppp
```

This command enables PPP on an interface.

## Enabling PAP or CHAP

To enable CHAP or PAP authentication on an interface configured for PPP encapsulation, use the following command in interface configuration mode:

```
Device(config-if)# ppp authentication chap pap
```

Defines the authentication protocols supported and the order in which they are used. In this command, *protocol1, protocol2* represent the following protocols: CHAP, MS-CHAP, and PAP. PPP authentication is attempted first using the first authentication method, which is *protocol1*. If *protocol1* is unable to establish authentication, the next configured protocol is used to negotiate authentication.

If you configure **ppp authentication chap** on an interface, all incoming calls on that interface that initiate a PPP connection will have to be authenticated using CHAP; likewise, if you configure **ppp authentication pap**, all incoming calls that start a PPP connection will have to be authenticated via PAP. If you configure **ppp authentication chap pap**, the access server will attempt to authenticate all incoming calls that start a

PPP session with CHAP. If the remote device does not support CHAP, the access server will try to authenticate the call using PAP. If the remote device does not support either CHAP or PAP, authentication will fail and the call will be dropped. If you configure **ppp authentication pap chap**, the access server will attempt to authenticate all incoming calls that start a PPP session with PAP. If the remote device does not support PAP, the access server will try to authenticate the call using CHAP. If the remote device does not support either protocol, authentication will fail and the call will be dropped. If you configure the **ppp authentication** command with the **callin** keyword, the access server will only authenticate the remote device if the remote device initiated the call.

Authentication method lists and the **one-time** keyword are only available if you have enabled AAA; these will not be available if you are using TACACS or extended TACACS. If you specify the name of an authentication method list with the **ppp authentication** command, PPP will attempt to authenticate the connection using the methods defined in the specified method list. If AAA is enabled and no method list is defined by name, PPP will attempt to authenticate the connection using the methods defined as the default. The **ppp authentication** command with the **one-time** keyword enables support for one-time passwords during authentication.

The **if-needed** keyword is only available if you are using TACACS or extended TACACS. The **ppp authentication**command with the **if-needed** keyword means that PPP will only authenticate the remote device via PAP or CHAP if they have not yet authenticated during the life of the current call. If the remote device authenticated via a standard login procedure and initiated PPP from the EXEC prompt, PPP will not authenticate via CHAP if **ppp authentication chap if-needed** is configured on the interface.

**Caution** If you use a *list-name* that has not been configured with the **aaa authentication ppp**command, you disable PPP on the line.

## Inbound and Outbound Authentication

PPP supports two-way authentication. Normally, when a remote device dials in to an access server, the access server requests that the remote device prove that it is allowed access. This is known as inbound authentication. At the same time, the remote device can also request that the access server prove that it is who it says it is. This is known as outbound authentication. An access server also does outbound authentication when it initiates a call to a remote device.

## Enabling Outbound PAP Authentication

To enable outbound PAP authentication, use the following command in interface configuration mode:

```
Device(config-if)# ppp pap sent-username username1 password password1
```

The access server uses the username and password specified by the **ppp pap sent-username** command to authenticate itself whenever it initiates a call to a remote device or when it has to respond to a remote device's request for outbound authentication.

## Refusing PAP Authentication Requests

To refuse PAP authentication from peers requesting it, meaning that PAP authentication is disabled for all calls, use the following command in interface configuration mode:

```
Device(config-if)# ppp pap refuse
```

If the **refuse** keyword is not used, the device will not refuse any PAP authentication challenges received from the peer.

## Creating a Common CHAP Password

For remote CHAP authentication, you can configure your device to create a common CHAP secret password to use in response to challenges from an unknown peer. For example, if your device calls a rotary of devices (either from another vendor, or running an older version of the Cisco IOS XE software) to which a new (that is, unknown) device has been added. The **ppp chap password** command allows you to replace several username and password configuration commands with a single copy of this command on any dialer interface or asynchronous group interface.

To enable a device calling a collection of devices to configure a common CHAP secret password, use the following command in interface configuration mode:

```
Device(config-if)# ppp chap password secret
```

## Refusing CHAP Authentication Requests

To refuse CHAP authentication from peers requesting it, meaning that CHAP authentication is disabled for all calls, use the following command in interface configuration mode:

```
Device(config-if)# ppp chap refuse calling
```

If the **calling** keyword is used, the device will refuse to answer CHAP authentication challenges received from the peer, but will still require the peer to answer any CHAP challenges the device sends.

If outbound PAP has been enabled (using the **ppp pap sent-username** command), PAP is used as the authentication method in the refusal packet.

## Delaying CHAP Authentication Until Peer Authenticates

To specify that the device will not authenticate to a peer requesting CHAP authentication until after the peer has authenticated itself to the device, use the following command in interface configuration mode:

```
Device(config-if)# ppp chap wait secret
```

This command (which is the default) specifies that the device will not authenticate to a peer requesting CHAP authentication until the peer has authenticated itself to the device. The **no ppp chap wait**commandspecifies that the device will respond immediately to an authentication challenge.

# Using MS-CHAP

Microsoft Challenge Handshake Authentication Protocol (MS-CHAP) is the Microsoft version of CHAP and is an extension of RFC 1994. Like the standard version of CHAP, MS-CHAP is used for PPP authentication; in this case, authentication occurs between a PC using Microsoft Windows NT or Microsoft Windows 95 and a Cisco device or access server acting as a network access server.

MS-CHAP differs from the standard CHAP as follows:

- MS-CHAP is enabled by negotiating CHAP Algorithm 0x80 in LCP option 3, Authentication Protocol.
- The MS-CHAP Response packet is in a format designed to be compatible with Microsoft Windows NT 3.5 and 3.51, Microsoft Windows 95, and Microsoft LAN Manager 2.*x*. This format does not require the authenticator to store a clear or reversibly encrypted password.
- MS-CHAP provides an authenticator-controlled authentication retry mechanism.
- MS-CHAP provides an authenticator-controlled change password mechanism.
- MS-CHAP defines a set of “reason-for failure” codes returned in the Failure packet message field.

Depending on the security protocols you have implemented, PPP authentication using MS-CHAP can be used with or without AAA security services. If you have enabled AAA, PPP authentication using MS-CHAP can be used in conjunction with both TACACS+ and RADIUS. The table below lists the vendor-specific RADIUS attributes (IETF Attribute 26) that enable RADIUS to support MS-CHAP.

***Table 3: Vendor-Specific RADIUS Attributes for MS-CHAP***

| Vendor-ID Number | Vendor-Type Number | Vendor-Proprietary Attribute | Description |
|---|---|---|---|
| 311 | 11 | MSCHAP-Challenge | Contains the challenge sent by a network access server to an MS-CHAP user. It can be used in both Access-Request and Access-Challenge packets. |
| 211 | 11 | MSCHAP-Response | Contains the response value provided by a PPP MS-CHAP user in response to the challenge. It is only used in Access-Request packets. This attribute is identical to the PPP CHAP Identifier. |

## Domain Stripping

The AAA Broadcast Accounting feature allows accounting information to be sent to multiple AAA servers at the same time, that is, accounting information can be broadcast to one or more AAA servers simultaneously. This functionality allows you to send accounting information to private and public AAA servers. It also provides redundant billing information for voice applications.

The Domain Stripping feature allows domain stripping to be configured at the server group level.

Per-server group configuration overrides the global configuration. If domain stripping is not enabled globally, but it is enabled in a server group, then it is enabled only for that server group. Also, if virtual routing and forwarding (VRF)-specific domain stripping is configured globally and in a server group for a different VRF, domain stripping is enabled in both the VRFs. VRF configurations are taken from server-group configuration mode. If server-group configurations are disabled in global configuration mode but are available in server-group configuration mode, all configurations in server-group configuration mode are applicable.

After the domain stripping and broadcast accounting are configured, you can create separate accounting records as per the configurations.

If both **domain-stripping** and **directed-request** commands are enabled, domain stripping takes precedence and directed request functionality will not work.

# How to Configure Authentication

## Configuring Login Authentication Using AAA

The AAA security services facilitate a variety of login authentication methods. Use the **aaa authentication login**command to enable AAA authentication no matter which of the supported login authentication methods you decide to use. With the **aaa authentication login**command, you create one or more lists of authentication methods that are tried at login. These lists are applied using the **login authentication** line configuration command.

To configure login authentication by using AAA, use the following commands beginning in global configuration mode:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authentication login** {**default** \| *list-name*} *method1*[*method2*...]<br>**Example:**<br>`Device(config)# aaa authentication login default local` | Creates a local authentication list. |
| **Step 5** | **line** [**aux** \| **console** \| **tty** \| **vty**] **line-number** [**ending-line-number**]<br>**Example:**<br>`Device(config)# line vty 1` | Enters line configuration mode for the lines to which you want to apply the authentication list. |
| **Step 6** | **login authentication** {**default** \| *list-name*}<br>**Example:**<br>`Device(config-line)# login authentication default` | Applies the authentication list to a line or set of lines. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-line)# end` | Exits line configuration mode and returns to privileged EXEC mode. |

### What to do next

The *list-name* is a character string used to name the list you are creating. The method argument refers to the actual method the authentication algorithm tries. The additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

For example, to specify that authentication should succeed even if (in this example) the TACACS+ server returns an error, enter the following command:

```
Device(config)# aaa authentication login default group tacacs+ none
```

**Note** Because the **none** keyword enables *any* user logging in to successfully authenticate, it should be used only as a backup method of authentication.

To create a default list that is used when a named list is *not* specified in the **login authentication** command, use the **default** keyword followed by the methods that are to be used in default situations. The default method list is automatically applied to all interfaces.

For example, to specify RADIUS as the default method for user authentication during login, enter the following command:

```
Device(config)# aaa authentication login default group radius
```

# Configuring PPP Authentication Using AAA

The AAA security services facilitate a variety of authentication methods for use on serial interfaces running PPP. Use the **aaa authentication ppp**command to enable AAA authentication no matter which of the supported PPP authentication methods you decide to use.

To configure AAA authentication methods for serial lines using PPP, use the following commands in global configuration mode:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:** | Enables AAA. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# aaa new-model` | |
| **Step 4** | **aaa authentication ppp** {**default** \| *list-name*} *method1*[*method2...*]<br>**Example:**<br>`Device(config)# aaa authentication ppp-auth default local` | Creates a local authentication list. |
| **Step 5** | **interface** *interface-type interface-number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 0/1/0` | Enters interface configuration mode for the interface to which you want to apply the authentication list. |
| **Step 6** | **ppp authentication** {*protocol1* [*protocol2...*]} [**if-needed**] {**default** \| *list-name*} [**callin**] [**one-time**][**optional**]<br>**Example:**<br>`Device(config)# ppp authentication ms-chap ppp-auth` | Applies the authentication list to a line or set of lines. In this command, *protocol1* and *protocol2* represent the following protocols: CHAP, MS-CHAP, and PAP. PPP authentication is attempted first using the first authentication method, specified by *protocol1*. If *protocol1* is unable to establish authentication, the next configured protocol is used to negotiate authentication. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

### What to do next

With the **aaa authentication ppp**command, you create one or more lists of authentication methods that are tried when a user tries to authenticate via PPP. These lists are applied using the **ppp authentication** line configuration command.

To create a default list that is used when a named list is *not* specified in the **ppp authentication** command, use the **default** keyword followed by the methods you want used in default situations.

For example, to specify the local username database as the default method for user authentication, enter the following command:

```
Device(config)# aaa authentication ppp default local
```

The *list-name* is any character string used to name the list you are creating. The method argument refers to the actual method the authentication algorithm tries. The additional methods of authentication are used only

if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

For example, to specify that authentication should succeed even if (in this example) the TACACS+ server returns an error, enter the following command:

```
Device(config)# aaa authentication ppp default group tacacs+ none
```

**Note** Because **none** allows all users logging in to authenticate successfully, it should be used as a backup method of authentication.

# Configuring ARAP Authentication Using AAA

Using the **aaa authentication arap** command, you can create one or more lists of authentication methods that are tried when AppleTalk Remote Access Protocol (ARAP) users attempt to log in to the device. These lists are used with the **arap authentication** line configuration command.

Use the following commands starting in global configuration mode:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authentication arap**<br>**Example:**<br>`Device(config)# aaa authentication arap`<br>**Example:**<br>Enables authentication for ARAP users. | |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **line** *number*<br>**Example:**<br>`Device(config)# line 1` | (Optional) Changes to line configuration mode. |
| **Step 6** | Device(config-line)# **autoselect arap**<br>**Example:**<br>`Device(config-line)# auto-select arap` | (Optional) Enables autoselection of ARAP. |
| **Step 7** | **autoselect during-login**<br>**Example:**<br>`Device(config-line)# autoselect during-login` | (Optional) Starts the ARAP session automatically at user login. |
| **Step 8** | **arap authentication** *list-name*<br>**Example:**<br>`Device(config-line)# arap authentication arap-authen` | (Optional—not needed if **default** is used in the **aaa authentication arap** command) Enables TACACS+ authentication for ARAP on a line. |
| **Step 9** | **end**<br>**Example:**<br>`Device(config-line)# end` | Exits line configuration mode and returns to the privileged EXEC mode. |

### What to do next

The *list-name* is any character string used to name the list you are creating. The *method* argument refers to the actual list of methods the authentication algorithm tries, in the sequence entered.

To create a default list that is used when a named list is *not* specified in the **arap authentication** command, use the **default** keyword followed by the methods you want to use in default situations.

The additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

**Note** Because **none** allows all users logging in to be authenticated, it should be used as a backup method of authentication.

For example, to create a default AAA authentication method list used with ARAP, use the following command:

```
Device(config)# aaa authentication arap default if-needed none
```

To create the same authentication method list for ARAP and name the list *MIS-access*, use the following command:

```
Device(config)# aaa authentication arap MIS-access if-needed none
```

# Configuring NASI Authentication Using AAA

Using the **aaa authentication nasi** command, you can create one or more lists of authentication methods that are tried when NetWare Asynchronous Services Interface (NASI) users attempt to log in to the device. These lists are used with the **nasi authentication line** configuration command.

To configure NASI authentication using AAA, use the following commands starting in global configuration mode:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA globally. |
| **Step 4** | **aaa authentication nasi**<br>**Example:**<br>`Device(config)# aaa authentication nasi` | Enables authentication for NASI users. |
| **Step 5** | **line** *number*<br>**Example:**<br>`Device(config)# line 4` | (Optional--not needed if **default** is used in the **aaa authentication nasi** command.) Enters line configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **nasi authentication** *list-name*<br>**Example:**<br>`Device(config-line)# nasi authentication nasi-authen` | (Optional--not needed if **default** is used in the **aaa authentication nasi** command.) Enables authentication for NASI on a line. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-line)# end` | Exits line configuration mode and returns to the privileged EXEC mode. |

### What to do next

The *list-name* is any character string used to name the list you are creating. The *method* argument refers to the actual list of methods that the authentication algorithm tries, in the sequence entered.

To create a default list that is used when a named list is *not* specified in the **aaa authentication nasi** command, use the **default** keyword followed by the methods you want to use in default situations.

The additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

**Note** Because **none** allows all users logging in to be authenticated, it should be used as a backup method of authentication.

# Preventing an Access Request with a Blank Username from Being Sent to the RADIUS Server

The following configuration steps provide the ability to prevent an Access Request with a blank username from being sent to the RADIUS server. This functionality ensures that unnecessary RADIUS server interaction is avoided, and RADIUS logs are kept short.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# configure terminal` | Enables AAA globally. |
| **Step 4** | **aaa authentication suppress null-username**<br>**Example:**<br>`Device(config)# aaa authentication suppress null-username` | Prevents an Access Request with a blank username from being sent to the RADIUS server. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring Message Banners for AAA Authentication

AAA supports the use of configurable, personalized login and failed-login banners. You can configure message banners that will be displayed when a user logs in to the system to be authenticated using AAA and when, for whatever reason, authentication fails.

## Configuring a Login Banner

To configure a banner that is displayed when a user logs in (replacing the default message for login), perform the following task:

### Before you begin

To create a login banner, you must configure a delimiting character that notifies the system that the following text string must be displayed as the banner, and then the text string itself. The delimiting character is repeated at the end of the text string to signify the end of the banner. The delimiting character can be any single character in the extended ASCII character set, but once defined as the delimiter, that character cannot be used in the text string for the banner.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:** | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device> enable` | |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authentication banner** *delimiter string delimiter*<br>**Example:**<br>`Device(config)# aaa authentication banner *Unauthorized use is prohibited.*` | Creates a personalized login banner. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

## Configuring a Failed-Login Banner

To configure a message that is displayed when a user login fails (replacing the default message for failed login), perform the following task:

### Before you begin

To create a failed-login banner, you must configure a delimiting character, which notifies the system that the following text string must be displayed as the banner, and then configure the text string itself. The delimiting character is repeated at the end of the text string to signify the end of the failed-login banner. The delimiting character can be any single character in the extended ASCII character set, but once defined as the delimiter, that character cannot be used in the text string making up the banner.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authentication fail-message** *delimiter string delimiter*<br>**Example:**<br>`Device(config)# aaa authentication fail-message *Failed login. Try again.*` | Creates a message to be displayed when a user login fails. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring AAA Packet of Disconnect

Packet of disconnect (POD) terminates connections on the network access server (NAS) when particular session attributes are identified. By using session information obtained from AAA, the POD client residing on a UNIX workstation sends disconnect packets to the POD server running on the network access server. The NAS terminates any inbound user session with one or more matching key attributes. It rejects requests when required fields are missing or when an exact match is not found.

To configure POD, perform the following tasks in global configuration mode:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | **aaa accounting network default start-stop radius**<br>**Example:**<br>`Device(config)# aaa accounting network default start-stop radius` | Enables AAA accounting records. |
| **Step 4** | **aaa accounting delay-start**<br>**Example:**<br>`Device(config)# aaa accounting delay-start` | (Optional) Delays generation of the start accounting record until the Framed-IP-Address is assigned, allowing its use in the POD packet. |
| **Step 5** | **aaa pod server server-key** *string*<br>**Example:**<br>`Device(config)# aaa pod server server-key xyz123` | Enables POD reception. |
| **Step 6** | **radius server** *name* non-standard<br>**Example:**<br>`Device(config)# radius server radser` | Configures a RADIUS server and enters RADIUS server configuration mode. |
| **Step 7** | **address** {**ipv4** \| **ipv6**} *hostname*<br>**Example:**<br>`Device(config-radius-server)# address ipv4 radius-host` | Configures a RADIUS host. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config-radius-server)# end` | Exits RADIUS server configuration mode and returns to privileged EXEC mode. |

# Configuring Double Authentication

To configure double authentication, perform the following steps:

1. Enable AAA by using the **aaa-new model**global configuration command.

2. Use the **aaa authentication** command to configure your network access server to use login and PPP authentication method lists, then apply those method lists to the appropriate lines or interfaces.
3. Use the **aaa authorization** command to configure AAA network authorization at login.
4. Configure security protocol parameters (for example, RADIUS or TACACS+).
5. Use access control list AV pairs on the security server that the user can connect to the local host only by establishing a Telnet connection.
6. (Optional) Configure the **access-profile**command as an autocommand. If you configure the autocommand, remote users will not have to manually enter the **access-profile** command to access authorized rights associated with their personal user profile.

**Note** If the **access-profile** command is configured as an autocommand, users will still have to Telnet to the local host and log in to complete double authentication.

Follow these rules when creating the user-specific authorization statements (These rules relate to the default behavior of the **access-profile** command):

- Use valid AV pairs when configuring access control list AV pairs on the security server.
- For remote users to use the interface's existing authorization (that which existed prior to the second stage authentication/authorization), but you want them to have different access control lists (ACLs), you should specify *only* ACL AV pairs in the user-specific authorization definition. This might be desirable if you set up a default authorization profile to apply to the remote host, but want to apply specific ACLs to specific users.
- When these user-specific authorization statements are later applied to the interface, they can either be *added to* the existing interface configuration or they can *replace* the existing interface configuration--depending on which form of the **access-profile** command is used to authorize the user. You should understand how the **access-profile** command works before configuring the authorization statements.
- If you will be using ISDN or Multilink PPP, you must also configure virtual templates at the local host.

## Enabling Automated Double Authentication

You can make the double authentication process easier for users by implementing automated double authentication. Automated double authentication provides all of the security benefits of double authentication, but offers a simpler, more user-friendly interface for remote users. With double authentication, a second level of user authentication is achieved when the user Telnets to the network access server or router and enters a username and password. With automated double authentication, the user does not have to Telnet to the network access server; instead the user responds to a dialog box that requests a username and password or personal identification number (PIN). To use the automated double authentication feature, the remote user hosts must be running a companion client application.

**Note** Automated double authentication, like the existing double authentication feature, is for Multilink PPP ISDN connections only. Automated double authentication cannot be used with other protocols such as X.25 or SLIP.

Automated double authentication is an enhancement to the existing double authentication feature. To configure automated double authentication, you must first configure double authentication by completing the following steps:

1. Enable AAA by using the **aaa-new model** global configuration command.
2. Use the **aaa authentication** command to configure your network access server to use login and PPP authentication method lists, then apply those method lists to the appropriate lines or interfaces.
3. Use the **aaa authorization** command to configure AAA network authorization at login.
4. Configure security protocol parameters (for example, RADIUS or TACACS+).
5. Use access control list AV pairs on the security server that the user can connect to the local host only by establishing a Telnet connection.
6. Configure the **access-profile**command as an autocommand. If you configure the autocommand, remote users will not have to manually enter the **access-profile** command to access authorized rights associated with their personal user profile.

**Note** If the **access-profile** command is configured as an autocommand, users will still have to Telnet to the local host and log in to complete double authentication.

Follow these rules when creating the user-specific authorization statements (These rules relate to the default behavior of the **access-profile** command):

- Use valid AV pairs when configuring access control list AV pairs on the security server.
- If you want remote users to use the interface's existing authorization (that which existed prior to the second stage authentication/authorization), but you want them to have different access control lists (ACLs), you should specify *only* ACL AV pairs in the user-specific authorization definition. This might be desirable if you set up a default authorization profile to apply to the remote host, but want to apply specific ACLs to specific users.
- When these user-specific authorization statements are later applied to the interface, they can either be *added to* the existing interface configuration, or *replace* the existing interface configuration--depending on which form of the **access-profile** command is used to authorize the user. You should understand how the **access-profile** command works before configuring the authorization statements.
- If you will be using ISDN or Multilink PPP, you must also configure virtual templates at the local host.

## Configuring Automated Double Authentication

To configure automated double authentication, perform the following task:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:** | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device> enable` | |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip trigger-authentication** [**timeout** *seconds*] [**port** *number*]<br>**Example:**<br>`Device(config)# ip trigger-authentication timeout 120` | Enables automation of double authentication. |
| **Step 4** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/17` | Configures an interface and enter the interface configuration mode. |
| **Step 5** | **ip trigger-authentication**<br>**Example:**<br>`Device(config-if)# ip rigger-authentication` | Applies automated double authentication to the interface. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

## Troubleshooting Automated Double Authentication

To troubleshoot automated double authentication, use the following commands in privileged EXEC mode:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **show ip trigger-authentication**<br>**Example:**<br>`Device# show ip trigger-authentication` | Displays the list of remote hosts for which automated double authentication has been attempted (successfully or unsuccessfully). |
| **Step 3** | **clear ip trigger-authentication**<br>**Example:**<br>`Device# clear ip trigger-authentication` | Clears the list of remote hosts for which automated double authentication has been attempted. (This clears the table displayed by the **show ip trigger-authentication** command.) |
| **Step 4** | **debug ip trigger-authentication**<br>**Example:**<br>`Device# debug ip trigger-authentication` | Displays debug output related to automated double authentication. |

# Configuring Domain Stripping at the Server Group Level

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa group server radius** *server-name*<br>**Example:**<br>`Device(config)# aaa group server radius rad1` | Adds the RADIUS server and enters server group RADIUS configuration mode.<br>• The *server-name* argument specifies the RADIUS server group name. |
| **Step 4** | **domain-stripping** [**strip-suffix** *word*] [**right-to-left** ] [**prefix-delimiter** *word*] [**delimiter** *word*]<br>**Example:**<br>`Device(config-sg-radius)# domain-stripping delimiter username@example.com` | Configures domain stripping at the server group level. |
| **Step 5** | **end**<br>**Example:** | Exits server group RADIUS configuration mode and returns to the privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-sg-radius)# end` | |

# Configuring Non-AAA Authentication Methods

## Configuring Line Password Protection

This task is used to provide access control on a terminal line by entering the password and establishing password checking.

**Note** If you configure line password protection and then configure TACACS or extended TACACS, the TACACS username and password take precedence over line passwords. If you have not yet implemented a security policy, we recommend that you use AAA.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **line** [**aux** \| **console** \| **tty** \| **vty**] *line-number* [*ending-line-number*]<br>**Example:**<br>`Device(config)# line console 0` | Enters line configuration mode. |
| **Step 4** | **password** *password*<br>**Example:**<br>`Device(config-line)# secret word` | Assigns a password to a terminal or other device on a line. The password checker is case sensitive and can include spaces; for example, the password “Secret” is different from the password “secret,” and “two words” is an acceptable password. |
| **Step 5** | **login**<br>**Example:**<br>`Device(config-line)# login` | Enables password checking at login.<br>You can disable line password verification by disabling password checking by using the **no** version of this command. |

| | Command or Action | Purpose |
|---|---|---|
| | | **Note**<br>The **login** command only changes username and privilege level but it does not execute a shell; therefore autocommands will not be executed. To execute autocommands under this circumstance, you need to establish a Telnet session back into the device (loop-back). Make sure that the device has been configured for secure Telnet sessions if you choose to implement autocommands this way. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-line)# end` | Exits line configuration mode and returns to privileged EXEC mode. |

## Establishing Username Authentication

You can create a username-based authentication system, which is useful in the following situations:

- To provide a TACACS-like username and encrypted password-authentication system for networks that cannot support TACACS
- To provide special-case logins: for example, access list verification, no password verification, autocommand execution at login, and “no escape” situations

To establish username authentication, perform the following task:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | Do one of the following:<br>• **username** *name* [**nopassword** \| **password** *password* \| **password** *encryption-type encrypted password*]<br>• **username** *name* [**access-class** *number*] | Establishes username authentication with encrypted passwords.<br>or<br>(Optional) Establishes username authentication by access list. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device(config)# username superuser password superpassword password 7 encrypted-password`<br>`Device(config)# username user1 access-class access-user` | |
| **Step 4** | **username** *name* [**privilege** *level*]<br>**Example:**<br>`Device(config)# username user1 privilege 5` | (Optional) Sets the privilege level for the user. |
| **Step 5** | **username** *name* [**autocommand** *command*]<br>**Example:**<br>`Device(config)# username user1 autocommand show users` | (Optional) Specifies a command to be executed automatically. |
| **Step 6** | **username** *name* [**noescape**] [**nohangup**]<br>**Example:**<br>`Device(config)# username user1 noescapee` | (Optional) Sets a “no escape” login environment. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

### What to do next

The keyword **noescape** prevents users from using escape characters on the hosts to which they are connected. The **nohangup** feature does not disconnect after using the autocommand.

**Caution** Passwords will be displayed in clear text in your configuration unless you enable the **service password-encryption** command.

## Defining PPP Authentication Using MS-CHAP

To define PPP authentication using MS-CHAP, use the following commands in interface configuration mode:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **encapsulation ppp**<br>**Example:**<br>`Device(config)# encapsulation ppp` | Enables PPP encapsulation. |
| **Step 4** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/2` | Configures an interface and enters interface configuration mode. |
| **Step 5** | **ppp authentication ms-chap** [**if-needed**] [*list-name* \| **default**] [**callin**] [**one-time**]<br>**Example:**<br>`Device(config-if)# ppp authentication ms-chap default callin` | Defines PPP authentication using MS-CHAP. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

### What to do next

If you configure **ppp authentication ms-chap** on an interface, all incoming calls on that interface that initiate a PPP connection will have to be authenticated using MS-CHAP. If you configure the **ppp authentication** command with the **callin** keyword, the access server will only authenticate the remote device if the remote device initiated the call.

Authentication method lists and the **one-time** keyword are only available if you have enabled AAA--they will not be available if you are using TACACS or extended TACACS. If you specify the name of an authentication method list with the **ppp authentication** command, PPP will attempt to authenticate the connection using the methods defined in the specified method list. If AAA is enabled and no method list is defined by name, PPP will attempt to authenticate the connection using the methods defined as the default.

The **ppp authentication** command with the **one-time** keyword enables support for one-time passwords during authentication.

The **if-needed** keyword is only available if you are using TACACS or extended TACACS. The **ppp authentication**command with the **if-needed** keyword means that PPP will only authenticate the remote device via MS-CHAP if that device has not yet authenticated during the life of the current call. If the remote device authenticated through a standard login procedure and initiated PPP from the EXEC prompt, PPP will not authenticate through MS-CHAP if **ppp authentication chap if-needed** is configured.

**Note** If PPP authentication using MS-CHAP is used with username authentication, you must include the MS-CHAP secret in the local username/password database.

# Configuration Examples for Authentication

## Example: Configuring Method Lists

Suppose the system administrator has decided on a security solution where all interfaces will use the same authentication methods to authenticate PPP connections. In the RADIUS group, R1 is contacted first for authentication information, then if there is no response, R2 is contacted. If R2 does not respond, T1 in the TACACS+ group is contacted; if T1 does not respond, T2 is contacted. If all designated servers fail to respond, authentication falls to the local username database on the access server itself. To implement this solution, the system administrator would create a default method list by entering the following command:

```
Device> enable
Device# configure terminal
Device(config)# aaa authentication ppp default group radius group tacacs+ local
Device(config)# exit
```

In this example, “default” is the name of the method list. The protocols included in this method list are listed after the name, in the order they are to be queried. The default list is automatically applied to all interfaces.

When a remote user attempts to dial in to the network, the network access server first queries R1 for authentication information. If R1 authenticates the user, it issues a PASS response to the network access server and the user is allowed to access the network. If R1 returns a FAIL response, the user is denied access and the session is terminated. If R1 does not respond, then the network access server processes that as an ERROR and queries R2 for authentication information. This pattern would continue through the remaining designated methods until the user is either authenticated or rejected, or until the session is terminated.

It is important to remember that a FAIL response is significantly different from an ERROR. A FAIL means that the user has not met the criteria contained in the applicable authentication database to be successfully authenticated. Authentication ends with a FAIL response. An ERROR means that the security server has not responded to an authentication query. Because of this, no authentication has been attempted. Only when an ERROR is detected will AAA select the next authentication method defined in the authentication method list.

Suppose the system administrator wants to apply a method list only to a particular interface or set of interfaces. In this case, the system administrator creates a named method list and then applies this named list to the applicable interfaces. The following example shows how the system administrator can implement an authentication method that will be applied only to interface 3:

```
Device> enable
Device# configure terminal
Device(config)# Device(config)#
Device(config)# aaa authentication ppp server-group1 group radius group tacacs+ local none
Device(config)# interface gigabitethernet 1/0/3
Device(config-if)#  ppp authentication chap server-group1
Device(config-if)# end
```

In this example, “apple” is the name of the method list, and the protocols included in this method list are listed after the name in the order in which they are to be performed. After the method list has been created, it is applied to the appropriate interface. Note that the method list name (apple) in both the AAAand PPP authentication commands must match.

In the following example, the system administrator uses server groups to specify that only R2 and T2 are valid servers for PPP authentication. To do this, the administrator must define specific server groups whose members are R2 (172.16.2.7) and T2 (172.16.2.77), respectively. In this example, the RADIUS server group “rad2only” is defined as follows using the **aaa group server** command:

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius rad2only
Device(config-sg-radius)# server 172.16.2.7
Device(config-sg-radius)# end
```

The TACACS+ server group “tac2only” is defined as follows using the **aaa group server** command:

```
Device> enable
Device# configure terminal
Device(config)# aaa group server tacacs+ tac2only
Device(config-sg-tacacs)# server 172.16.2.77
Device(config-sg-tacacs)# end
```

The administrator then applies PPP authentication using the server groups. In this example, the default methods list for PPP authentication follows this order: **group rad2only**, **group tac2only**, and **local**:

```
Device> enable
Device# configure terminal
Device(config)# aaa authentication ppp default group rad2only group tac2only local
Device(config)# exit
```

If a method list is configured under VTY lines, the corresponding method list must be added to AAA. The following example shows how to configure a method list under a VTY line:

```
Device> enable
Device# configure terminal
Device(config)# line vty 0 4
Device(config-line)# authorization commands 15 auth1
Device(config-line)# exit
```

The following example shows how to configure a method list in AAA:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authorization commands 15 auth1 group tacacs+
Device(config)# exit
```

If no method list is configured under VTY lines, the default method list must be added to AAA. The following example shows a VTY configuration without a method list:

```
Device> enable
Device# configure terminal
Device(config)# line vty 0 4
Device(config-line)# end
```

The following example shows how to configure the default method list:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authorization commands 15 default group tacacs+
Device(config)# exit
```

# Example: RADIUS Authentication

This section provides two sample configurations using RADIUS.

The following example shows how to configure the router to authenticate and authorize using RADIUS:

```
Device> enable
Device# configure terminal
Device(config)# aaa authentication login radius-login group radius local
Device(config)# aaa authentication ppp radius-ppp if-needed group radius
Device(config)# aaa authorization exec default group radius if-authenticated
Device(config)# aaa authorization network default group radius
Device(config)# line 3
Device(config-line)# login authentication radius-login
Device(config-line)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# ppp authentication radius-ppp
Device(config-if)# end
```

The lines in this sample RADIUS authentication and authorization configuration are defined as follows:

- The aaa authentication login radius-login group radius local command configures the router to use RADIUS for authentication at the login prompt. If RADIUS returns an error, the user is authenticated using the local database.
- The aaa authentication ppp radius-ppp if-needed group radius command configures the Cisco IOS XE software to use PPP authentication using CHAP or PAP if the user has not already logged in. If the EXEC facility has authenticated the user, PPP authentication is not performed.
- The aaa authorization exec default group radius if-authenticated command queries the RADIUS database for information that is used during EXEC authorization, such as autocommands and privilege levels, but only provides authorization if the user has successfully authenticated.
- The aaa authorization network default group radius command queries RADIUS for network authorization, address assignment, and other access lists.
- The **login authentication radius-login** command enables the radius-login method list for line 3.
- The **ppp authentication radius-ppp** command enables the radius-ppp method list for serial interface 0.

The following example shows how to configure the router to prompt for and verify a username and password, authorize the user's EXEC level, and specify it as the method of authorization for privilege level 2. In this example, if a local username is entered at the username prompt, that username is used for authentication.

If the user is authenticated using the local database, EXEC authorization using RADIUS will fail because no data is saved from the RADIUS authentication. The method list also uses the local database to find an autocommand. If there is no autocommand, the user becomes the EXEC user. If the user then attempts to issue commands that are set at privilege level 2, TACACS+ is used to attempt to authorize the command.

```
Device> enable
Device# configure terminal
Device(config)# aaa authentication login default group radius local
Device(config)# aaa authorization exec default group radius local
Device(config)# aaa authorization command 2 default group tacacs+ if-authenticated
Device(config)# radius server radserver
Device(config-sg-radius)# address ipv4 10.2.3.1
Device(config-sg-radius)# exit
Device(config)# radius-server attribute 44 include-in-access-req
Device(config)# radius-server attribute 8 include-in-access-req
Device(config)# end
```

The lines in this sample RADIUS authentication and authorization configuration are defined as follows:

- The aaa authentication login default group radius local command specifies that the username and password are verified by RADIUS or, if RADIUS is not responding, by the router's local user database.
- The aaa authorization exec default group radius local command specifies that RADIUS authentication information be used to set the user's EXEC level if the user authenticates with RADIUS. If no RADIUS information is used, this command specifies that the local user database be used for EXEC authorization.
- The aaa authorization command 2 default group tacacs+ if-authenticated command specifies TACACS+ authorization for commands set at privilege level 2, if the user has already successfully authenticated.
- The radius-server attribute 44 include-in-access-req command sends RADIUS attribute 44 (Acct-Session-ID) in access-request packets.
- The radius-server attribute 8 include-in-access-req command sends RADIUS attribute 8 (Framed-IP-Address) in access-request packets.

# Example: TACACS Authentication

The following example shows how to configure TACACS+ as the security protocol to be used for PPP authentication:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication ppp test group tacacs+ local
Device(config)# interface gigabitethernet 1/1/2
Device(config-if)# ppp authentication chap pap test
Device(config-if)# exit
Device(config)# tacacs server server1
Device(config-server-tacacs)# address ipv4 192.0.2.3
Device(config-server-tacacs)# key key1
Device(config-server-tacacs)# end
```

The lines in this sample TACACS+ authentication configuration are defined as follows:

- The **aaa new-model** command enables the AAA security services.
- The **aaa authentication** command defines a method list, "test," to be used on serial interfaces running PPP. The keywords **group tacacs+** means that authentication will be done through TACACS+. If

TACACS+ returns an ERROR of some sort during authentication, the keyword **local** indicates that authentication will be attempted using the local database on the network access server.

- The **interface** command selects the line.
- The **ppp authentication** command applies the test method list to this line.
- The **address ipv4** command identifies the TACACS+ daemon as having an IP address of 192.0.2.3.
- The **key** command defines the shared encryption key to be "key1."

The following example shows how to configure AAA authentication for PPP:

```
Device(config)# aaa authentication ppp default if-needed group tacacs+ local
```

In this example, the keyword **default** means that PPP authentication is applied by default to all interfaces. The **if-needed** keyword means that if the user has already authenticated by going through the ASCII login procedure, then PPP is not necessary and can be skipped. If authentication is needed, the keywords **group tacacs+** means that authentication will be done through TACACS+. If TACACS+ returns an ERROR of some sort during authentication, the keyword **local** indicates that authentication will be attempted using the local database on the network access server.

The following example shows how to create the same authentication algorithm for PAP, but it calls the method list "MIS-access" instead of "default":

```
Device> enable
Device# configure terminal
Device(config)# aaa authentication ppp MIS-access if-needed group tacacs+ local
Device(config)# interface gigabitethernet 1/1/2
Device(config)# ppp authentication pap MIS-access
Device(config)# end
```

In this example, because the list does not apply to any interfaces (unlike the default list, which applies automatically to all interfaces), the administrator must select interfaces to which this authentication scheme should apply by using the **interface** command. The administrator must then apply this method list to those interfaces by using the **ppp authentication** command.

# Example: Kerberos Authentication

To specify Kerberos as the login authentication method, use the following command:

```
Device> enable
Device# configure terminal
Device(config)# aaa authentication login default krb5
Device(config)# end
```

To specify Kerberos authentication for PPP, use the following command:

```
Device> enable
Device# configure terminal
Device(config)# aaa authentication ppp default krb5
Device(config)# end
```

# Example: AAA Scalability

The following example shows a general security configuration using AAA with RADIUS as the security protocol. In this example, the network access server is configured to allocate 16 background processes to handle AAA requests for PPP.

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# radius server radserver
Device(config-sg-radius)# address ipv4 radius-host
Device(config-sg-radius)# key myRaDiUSpassWoRd
Device(config-sg-radius)# exit
Device(config)# radius-server configure-nas
Device(config)# username root password ALongPassword
Device(config)# aaa authentication ppp dialins group radius local
Device(config)# aaa authentication login admins local
Device(config)# aaa authorization network default group radius local
Device(config)# aaa accounting network default start-stop group radius
Device(config)# aaa processes 16
Device(config)# line 1 16
Device(config-line)# autoselect ppp
Device(config-line)# autoselect during-login
Device(config-line)# login authentication admins
Device(config-line)# modem dialin
Device(config-line)# exit
Device(config)# interface gigabitethernet 1/2/0
Device(config-if)# group-range 1 16
Device(config-if)# encapsulation ppp
Device(config-if)# ppp authentication pap dialins
Device(config-if)# end
```

The lines in this sample RADIUS AAA configuration are defined as follows:

- The **aaa new-model** command enables AAA network security services.
- The **address ipv4** {*hostname* | *host-address*} command defines the name of the RADIUS server host.
- The **key** command defines the shared secret text string between the network access server and the RADIUS server host.
- The **radius-server configure-nas** command defines that the Cisco router or access server will query the RADIUS server for static routes and IP pool definitions when the device first starts up.
- The **username** command defines the username and password to be used for the PPP Password Authentication Protocol (PAP) caller identification.
- The **aaa authentication ppp dialins group radius local** command defines the authentication method list "dialins," which specifies that RADIUS authentication, then (if the RADIUS server does not respond) local authentication will be used on serial lines using PPP.
- The **aaa authentication login admins local** command defines another method list, "admins," for login authentication.
- The **aaa authorization network default group radius local** command is used to assign an address and other network parameters to the RADIUS user.
- The **aaa accounting network default start-stop group radius** command tracks PPP usage.
- The **aaa processes** command allocates 16 background processes to handle AAA requests for PPP.

- The **line** command switches the configuration mode from global configuration to line configuration and identifies the specific lines being configured.
- The **autoselect ppp** command allows a PPP session to start up automatically on these selected lines.
- The **autoselect during-login** command is used to display the username and password prompt without pressing the Return key. After the user logs in, the autoselect function (in this case, PPP) begins.
- The **login authentication admins** command applies the "admins" method list for login authentication.
- The **modem dialin** command configures modems attached to the selected lines to only accept incoming calls.
- The **interface group-async** command selects and defines an asynchronous interface group.
- The **group-range** command defines the member asynchronous interfaces in the interface group.
- The **encapsulation ppp** command sets PPP as the encapsulation method used on the specified interfaces.
- The **ppp authentication pap dialins**command applies the "dialins" method list to the specified interfaces.

# Example: Configuring Login and Failed-Login Banners for AAA Authentication

The following example shows how to configure a login banner that is displayed when a user logs in to the system, (in this case, the phrase "Unauthorized Access Prohibited"). The asterisk (*) is used as the delimiting character. RADIUS is specified as the default login authentication method.

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication banner *Unauthorized Access Prohibited*
Device(config)# aaa authentication login default group radius
Device(config)# end
```

This configuration displays the following login banner:

```
Unauthorized Access Prohibited
Username:
```

The following example shows how to configure a failed-login banner that is displayed when a user tries to log in to the system and fails, (in this case, the phrase "Failed login. Try again"). The asterisk (*) is used as the delimiting character. RADIUS is specified as the default login authentication method.

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication banner *Unauthorized Access Prohibited*
Device(config)# aaa authentication fail-message *Failed login. Try again.*
Device(config)# aaa authentication login default group radius
Device(config)# end
```

This configuration displays the following login and failed-login banner:

```
Unauthorized Access Prohibited
Username:
Password:
Failed login. Try again.
```

# Example: AAA Packet of Disconnect Server Key

The following example shows how to configure POD (packet of disconnect), which terminates connections on the network access server (NAS) when particular session attributes are identified.

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication ppp default radius
Device(config)# aaa accounting network default start-stop radius
Device(config)# aaa accounting delay-start
Device(config)# aaa pod server server-key xyz123
Device(config)# radius server non-standard
Device(config-sg-radius)# address ipv4 10.2.1.1
Device(config-sg-radius)# key rad123
Device(config-sg-radius)# end
```

# Example: Double Authentication

The examples in this section illustrate possible configurations to be used with double authentication. Your configurations could differ significantly, depending on your network and security requirements.

**Note** These configuration examples include specific IP addresses and other specific information. This information is for illustration purposes only: your configuration will use different IP addresses, different usernames and passwords, and different authorization statements.

## Example: Configuration of the Local Host for AAA with Double Authentication

These two examples show how to configure a local host to use AAA for PPP and login authentication, and for network and EXEC authorization. An example each is shown for RADIUS and for TACACS+.

In both the examples, the first three lines configure AAA with a specific server as the AAA server. The next two lines configure AAA for PPP and login authentication, and the last two lines configure network and EXEC authorization. The last line is necessary only if the **access-profile** command will be executed as an autocommand.

The following example shows device configuration with a RADIUS AAA server:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# radius server radserver
Device(config-sg-radisu)# address ipv4 secureserver
Device(config-sg-radius)# key myradiuskey
Device(config-sg-radius)# exit
Device(config)# aaa authentication ppp default group radius
Device(config)# aaa authentication login default group radius
Device(config)# aaa authorization network default group radius
Device(config)# aaa authorization exec default group radius
Device(config)# end
```

The following example shows device configuration with a TACACS+ server:

```
Device> enable
Device# configure terminal
```

```
Device(config)# aaa new-model
Device(config)# tacacs server server1
Device(config-server-tacacs)# address ipv4 192.0.2.3
Device(config-server-tacacs)# key mytacacskey
Device(config-server-tacacs)# exit
Device(config)# aaa authentication ppp default group tacacs+
Device(config)# aaa authentication login default group tacacs+
Device(config)# aaa authorization network default group tacacs+
Device(config)# aaa authorization exec default group tacacs+
Device(config)# end
```

## Example: Configuration of the AAA Server for First-Stage PPP Authentication and Authorization

This example shows a configuration on the AAA server. A partial sample AAA configuration is shown for RADIUS.

TACACS+ servers can be configured similarly. (See the Complete Configuration with TACACS Example.)

This example defines authentication/authorization for a remote host named “hostx” that will be authenticated by CHAP in the first stage of double authentication. Note that the ACL AV pair limits the remote host to Telnet connections to the local host. The local host has the IP address 10.0.0.2.

The following example shows a partial AAA server configuration for RADIUS:

```
hostx   Password = “welcome”
        User-Service-Type = Framed-User,
        Framed-Protocol = PPP,
        cisco-avpair = “lcp:interface-config=ip unnumbered fastethernet 0”,
        cisco-avpair = “ip:inacl#3=permit tcp any 172.21.114.0 0.0.0.255 eq telnet”,
        cisco-avpair = “ip:inacl#4=deny icmp any any”,
        cisco-avpair = “ip:route#5=10.0.0.0 255.0.0.0”,
        cisco-avpair = “ip:route#6=10.10.0.0 255.0.0.0”,
        cisco-avpair = “ipx:inacl#3=deny any”
```

## Example: Configuration of the AAA Server for Second-Stage Per-User Authentication and Authorization

This section contains partial sample AAA configurations on a RADIUS server. These configurations define authentication and authorization for a user with the username “user1,” who will be user-authenticated in the second stage of double authentication.

TACACS+ servers can be configured similarly.

Three examples show sample RADIUS AAA configurations that could be used with each of the three forms of the **access-profile** command.

The first example shows a partial sample AAA configuration that works with the default form (no keywords) of the **access-profile** command. Note that only ACL AV pairs are defined. This example also sets up the **access-profile** command as an autocommand.

```
user1     Password = “welcome”
          User-Service-Type = Shell-User,
          cisco-avpair = “shell:autocmd=access-profile”
          User-Service-Type = Framed-User,
          Framed-Protocol = PPP,
          cisco-avpair = “ip:inacl#3=permit tcp any host 10.0.0.2 eq telnet”,
          cisco-avpair = “ip:inacl#4=deny icmp any any”
```

The second example shows a partial sample AAA configuration that works with the **access-profile merge** form of the **access-profile** command. This example also sets up the **access-profile merge**command as an autocommand.

```
user1     Password = "welcome"
          User-Service-Type = Shell-User,
          cisco-avpair = "shell:autocmd=access-profile merge"
          User-Service-Type = Framed-User,
          Framed-Protocol = PPP,
          cisco-avpair = "ip:inacl#3=permit tcp any any"
          cisco-avpair = "ip:route=10.0.0.0 255.255.0.0",
          cisco-avpair = "ip:route=10.1.0.0 255.255.0.0",
          cisco-avpair = "ip:route=10.2.0.0 255.255.0.0"
```

The third example shows a partial sample AAA configuration that works with the **access-profile replace** form of the **access-profile** command. This example also sets up the **access-profile replace**command as an autocommand.

```
user1     Password = "welcome"
          User-Service-Type = Shell-User,
          cisco-avpair = "shell:autocmd=access-profile replace"
          User-Service-Type = Framed-User,
          Framed-Protocol = PPP,
          cisco-avpair = "ip:inacl#3=permit tcp any any",
          cisco-avpair = "ip:inacl#4=permit icmp any any",
          cisco-avpair = "ip:route=10.10.0.0 255.255.0.0",
          cisco-avpair = "ip:route=10.11.0.0 255.255.0.0",
          cisco-avpair = "ip:route=10.12.0.0 255.255.0.0"
```

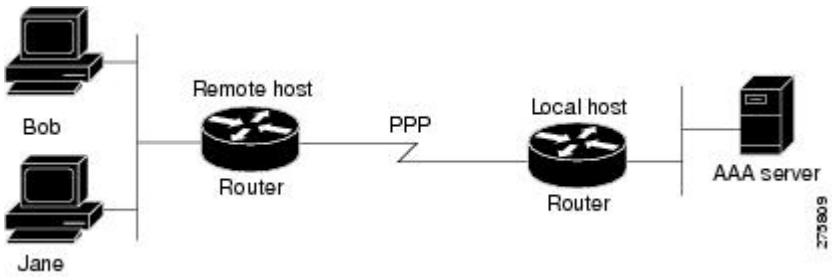
## Example: Complete Configuration with TACACS

This example shows TACACS+ authorization profile configurations both for the remote host (used in the first stage of double authentication) and for specific users (used in the second stage of double authentication).

This sample configuration shows authentication/authorization profiles on the TACACS+ server for the remote host "hostx" and for three users, with the usernames "user_default," "user_merge," and "user_replace." The configurations for these three usernames illustrate different configurations that correspond to the three different forms of the **access-profile** command. The three user configurations also illustrate setting up the autocommand for each form of the **access-profile** command.

The figure below shows the topology. The example that follows the figure shows a TACACS+ configuration file.

***Figure 3: Example Topology for Double Authentication***




This sample configuration shows authentication/authorization profiles on the TACACS+ server for the remote host "hostx" and for three users, with the usernames "user_default," "user_merge," and "user_replace."

```
key = "mytacacskey"
default authorization = permit
#----------------------------Remote Host (BRI)-------------------------
#
# This allows the remote host to be authenticated by the local host
# during fist-stage authentication, and provides the remote host
# authorization profile.
#
#-----------------------------------------------------------------------
user = hostx
{
    login = cleartext "welcome"
    chap = cleartext "welcome"
    service = ppp protocol = lcp {
                interface-config="ip unnumbered fastethernet 0"
    }
    service = ppp protocol = ip {
            # It is important to have the hash sign and some string after
            # it. This indicates to the NAS that you have a per-user
            # config.
            inacl#3="permit tcp any 172.21.114.0 0.0.0.255 eq telnet"
            inacl#4="deny icmp any any"
            route#5="10.0.0.0 255.0.0.0"
            route#6="10.10.0.0 255.0.0.0"
    }
    service = ppp protocol = ipx {
            # see previous comment about the hash sign and string, in protocol = ip
            inacl#3="deny any"
    }
}
#------------------- "access-profile" default user "only acls" ------------------
#
# Without arguments, access-profile removes any access-lists it can find
# in the old configuration (both per-user and per-interface), and makes sure
# that the new profile contains ONLY access-list definitions.
#
#-------------------------------------------------------------------------------
user = user_default
{
        login = cleartext "welcome"
        chap = cleartext "welcome"
        service = exec
        {
                # This is the autocommand that executes when user_default logs in.
                autocmd = "access-profile"
        }
        service = ppp protocol = ip {
                # Put whatever access-lists, static routes, whatever
                # here.
                # If you leave this blank, the user will have NO IP
                # access-lists (not even the ones installed prior to
                # this)!
                inacl#3="permit tcp any host 10.0.0.2 eq telnet"
                inacl#4="deny icmp any any"
        }
        service = ppp protocol = ipx {
                # Put whatever access-lists, static routes, whatever
                # here.
                # If you leave this blank, the user will have NO IPX
                # access-lists (not even the ones installed prior to
                # this)!
        }
}
```

```
#--------------------- “access-profile merge” user ---------------------------
#
# With the 'merge' option, first all old access-lists are removed (as before),
# but then (almost) all AV pairs are uploaded and installed. This will allow
# for uploading any custom static routes, sap-filters, and so on, that the user
# may need in his or her profile. This needs to be used with care, as it leaves
# open the possibility of conflicting configurations.
#
#-----------------------------------------------------------------------------
user = user_merge
{
        login = cleartext “welcome”
        chap = cleartext “welcome”
        service = exec
        {
                # This is the autocommand that executes when user_merge logs in.
                autocmd = “access-profile merge”
        }
        service = ppp protocol = ip
        {
                # Put whatever access-lists, static routes, whatever
                # here.
                # If you leave this blank, the user will have NO IP
                # access-lists (not even the ones installed prior to
                # this)!
                inacl#3=”permit tcp any any”
                route#2=”10.0.0.0 255.255.0.0"
                route#3=”10.1.0.0 255.255.0.0"
                route#4=”10.2.0.0 255.255.0.0"
        }
        service = ppp protocol = ipx
        {
                # Put whatever access-lists, static routes, whatever
                # here.
                # If you leave this blank, the user will have NO IPX
                # access-lists (not even the ones installed prior to
                # this)!
        }
}
#--------------------- “access-profile replace” user ----------------------------
#
# With the 'replace' option, ALL old configuration is removed and ALL new
# configuration is installed.
#
# One caveat: access-profile checks the new configuration for address-pool and
# address AV pairs. As addresses cannot be renegotiated at this point, the
# command will fail (and complain) when it encounters such an AV pair.
# Such AV pairs are considered to be “invalid” for this context.
#------------------------------------------------------------------------------
user = user_replace
{
        login = cleartex
t
“
welcome
”
        chap = cleartext “welcome”
        service = exec
        {
                # This is the autocommand that executes when user_replace logs in.
                autocmd = “access-profile replace”
        }
        service = ppp protocol = ip
        {
```

```
                # Put whatever access-lists, static routes, whatever
                # here.
                # If you leave this blank, the user will have NO IP
                # access-lists (not even the ones installed prior to
                # this)!
                inacl#3=”permit tcp any any”
                inacl#4=”permit icmp any any”
                route#2=”10.10.0.0 255.255.0.0"
                route#3=”10.11.0.0 255.255.0.0"
                route#4=”10.12.0.0 255.255.0.0"
        }
        service = ppp protocol = ipx
        {
                # put whatever access-lists, static routes, whatever
                # here.
                # If you leave this blank, the user will have NO IPX
                # access-lists (not even the ones installed prior to
                # this)!
        }
}
```

# Example: Automated Double Authentication

This example shows a complete configuration file with automated double authentication configured. The configuration commands that apply to automated double authentication are preceded by descriptions with a double asterisk (**).

```
Current configuration:
!
version 16.10
no service password-encryption
!
hostname myrouter
!
!
! **The following AAA commands are used to configure double authentication:
!
! **The following command enables AAA:
aaa new-model
! **The following command enables user authentication via the RADIUS AAA server:
!
aaa authentication login default none
aaa authentication ppp default group radius
! **The following command causes the remote user’s authorization profile to be
! downloaded from the AAA server to the router when required:
!
aaa authorization network default group radius
!
enable password mypassword
!
ip host blue 172.21.127.226
ip host green 172.21.127.218
ip host red 172.21.127.114
ip domain-name example.com
ip name-server 172.16.2.75
!
!
interface GigabitEthernet0/0/0
 ip address 172.21.127.186 255.255.255.248
 no ip route-cache
 no ip mroute-cache
```

```
 no keepalive
 ntp disable
 no cdp enable
!
interface Virtual-Template1
 ip unnumbered loopback0
 no ip route-cache
 no ip mroute-cache
!
! **The following command specifies that device authentication occurs via PPP CHAP:
 ppp authentication chap
!
router eigrp 109
 network 172.21.0.0
 no auto-summary
!
ip default-gateway 172.21.127.185
no ip classless
ip route 172.21.127.114 255.255.255.255 172.21.127.113
! **Virtual profiles are required for double authentication to work:
virtual-profile virtual-template 1
dialer-list 1 protocol ip permit
no cdp run
! **The following command defines where the TACACS+ AAA server is:
tacacs server server1
address ipv4 172.16.57.35
! **The following command defines the key to use with TACACS+ traffic (required):
key mytacacskey
snmp-server community public RO
!
line con 0
 exec-timeout 0 0
 login authentication console
line aux 0
 transport input all
line vty 0 4
 exec-timeout 0 0
 password lab
!
end
```

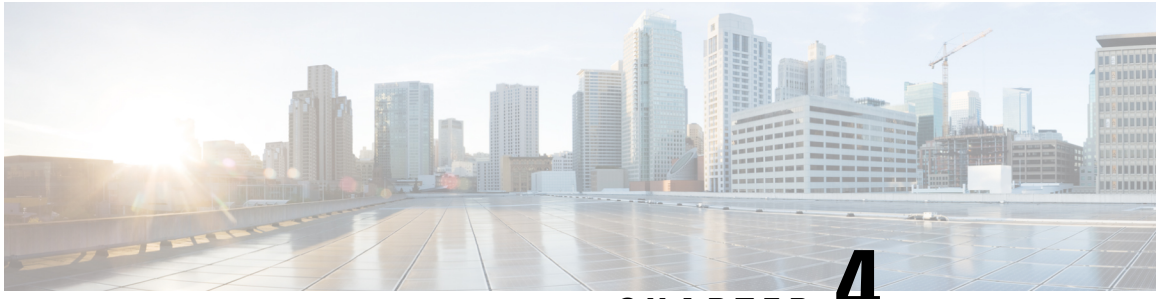
# Feature History for Configuring Authentication

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | AAA Authentication | Authentication provides a method to identify users, which includes the login and password dialog, challenge and response, messaging support, and encryption, depending on the selected security protocol. Authentication is the way a user is identified prior to being allowed access to the network and network services.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | AAA Authentication | Support for this feature was introduced on the C9500-24Y4C, C9500-32C, C9500-32QC, and C9500-48Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.



CHAPTER 4

# Configuring Authorization

AAA authorization enables you to limit the services available to a user. When AAA authorization is enabled, the network access server uses information retrieved from the user's profile, which is located either in the local user database or on the security server, to configure the user's session. Once this is done, the user will be granted access to a requested service only if the information in the user profile allows it.


- Prerequisites for Configuring Authorization, on page 77
- Information About Configuring Authorization, on page 78
- How to Configure Authorization, on page 81
- Configuration Examples for Authorization, on page 84
- Feature History for Configuring Authorization, on page 87


## Prerequisites for Configuring Authorization

Before configuring authorization using named method lists, you must first perform the following tasks:

- Enable authentication, authorization, and accounting (AAA) on your network access server.
- Configure AAA authentication. Authorization generally takes place after authentication and relies on authentication to work properly. For more information about AAA authentication, refer to the "Configuring Authentication" module.
- Define the characteristics of your RADIUS or TACACS+ security server if you are issuing RADIUS or TACACS+ authorization. For more information about configuring your Cisco network access server to communicate with your RADIUS security server, refer to the chapter "Configuring RADIUS". For more information about configuring your Cisco network access server to communicate with your TACACS+ security server, refer to the "Configuring TACACS+" module.
- Define the rights associated with specific users by using the **username** command if you are issuing local authorization.

# Information About Configuring Authorization

## Named Method Lists for Authorization

Method lists for authorization define the ways that authorization will be performed and the sequence in which these methods will be performed. A method list is simply a named list describing the authorization methods to be queried (such as RADIUS or TACACS+), in sequence. Method lists enable you to designate one or more security protocols to be used for authorization, thus ensuring a backup system in case the initial method fails. Cisco IOS XE software uses the first method listed to authorize users for specific network services; if that method fails to respond, the Cisco IOS XE software selects the next method listed in the list. This process continues until there is successful communication with a listed authorization method, or all methods defined are exhausted.

**Note** The Cisco IOS XE software attempts authorization with the next listed method only when there is no response from the previous method. If authorization fails at any point in this cycle--meaning that the security server or local username database responds by denying the user services--the authorization process stops and no other authorization methods are attempted.

Method lists are specific to the authorization type requested:

- Commands: Applies to the EXEC mode commands a user issues. Command authorization attempts authorization for all EXEC mode commands, including global configuration commands, associated with a specific privilege level.
- EXEC: Applies to the attributes associated with a user EXEC terminal session.
- Network: Applies to network connections. This can include a PPP, SLIP, or ARAP connection.
- Reverse Access: Applies to reverse Telnet sessions.

When you create a named method list, you are defining a particular list of authorization methods for the indicated authorization type.

Once defined, method lists must be applied to specific lines or interfaces before any of the defined methods will be performed. The only exception is the default method list (which is named “default”). If the **aaa authorization** command for a particular authorization type is issued without a named method list specified, the default method list is automatically applied to all interfaces or lines except those that have a named method list explicitly defined. (A defined method list overrides the default method list.) If no default method list is defined, local authorization takes place by default.

## AAA Authorization Methods

AAA supports five different methods of authorization:

- TACACS+: The network access server exchanges authorization information with the TACACS+ security daemon. TACACS+ authorization defines specific rights for users by associating attribute-value pairs, which are stored in a database on the TACACS+ security server, with the appropriate user.

- If-Authenticated: The user is allowed to access the requested function provided the user has been authenticated successfully.
- None: The network access server does not request authorization information; authorization is not performed over this line/interface.
- Local: The router or access server consults its local database, as defined by the **username** command, for example, to authorize specific rights for users. Only a limited set of functions can be controlled via the local database.
- RADIUS: The network access server requests authorization information from the RADIUS security server. RADIUS authorization defines specific rights for users by associating attributes, which are stored in a database on the RADIUS server, with the appropriate user.

**Note** With CSCuc32663, passwords and authorization logs are masked before being sent to the TACACS+, LDAP, or RADIUS security servers. Use the **aaa authorization commands visible-keys** command to send unmasked information to the TACACS+, LDAP, or RADIUS security servers.

## Authorization Methods

To have the network access server request authorization information via a TACACS+ security server, use the **aaa authorization** command with the **group tacacs+** *method* keyword. For more specific information about configuring authorization using a TACACS+ security server, refer to the chapter "Configuring TACACS+." For an example of how to enable a TACACS+ server to authorize the use of network services, including PPP and ARA, see the TACACS Authorization Examples.

To allow users to have access to the functions they request as long as they have been authenticated, use the **aaa authorization** command with the **if-authenticated** *method* keyword. If you select this method, all requested functions are automatically granted to authenticated users.

There may be times when you do not want to run authorization from a particular interface or line. To stop authorization activities on designated lines or interfaces, use the **none** *method* keyword. If you select this method, authorization is disabled for all actions.

To select local authorization, which means that the router or access server consults its local user database to determine the functions a user is permitted to use, use the **aaa authorization** command with the **local** *method* keyword. The functions associated with local authorization are defined by using the **username** global configuration command. For a list of permitted functions, refer to the chapter "Configuring Authentication."

To have the network access server request authorization via a RADIUS security server, use the **radius** *method* keyword. For more specific information about configuring authorization using a RADIUS security server, refer to the Configuring RADIUS chapter.

To have the network access server request authorization via a RADIUS security server, use the **aaa authorization** command with the **group radius** *method* keyword. For more specific information about configuring authorization using a RADIUS security server, refer to the chapter Configuring RADIUS. For an example of how to enable a RADIUS server to authorize services, see the RADIUS Authorization Example.

**Note** Authorization method lists for SLIP follow whatever is configured for PPP on the relevant interface. If no lists are defined and applied to a particular interface (or no PPP settings are configured), the default setting for authorization applies.

# Method Lists and Server Groups

A server group is a way to group existing RADIUS or TACACS+ server hosts for use in method lists. The figure below shows a typical AAA network configuration that includes four security servers: R1 and R2 are RADIUS servers, and T1 and T2 are TACACS+ servers. R1 and R2 make up the group of RADIUS servers. T1 and T2 make up the group of TACACS+ servers.

Using server groups, you can specify a subset of the configured server hosts and use them for a particular service. For example, server groups allow you to define R1 and R2 as separate server groups, and T1 and T2 as separate server groups. This means you can specify either R1 and T1 in the method list or R2 and T2 in the method list, which provides more flexibility in the way that you assign RADIUS and TACACS+ resources.

Server groups also can include multiple host entries for the same server, as long as each entry has a unique identifier. The combination of an IP address and a UDP port number creates a unique identifier, allowing different ports to be individually defined as RADIUS hosts providing a specific AAA service. In other words, this unique identifier enables RADIUS requests to be sent to different UDP ports on a server at the same IP address. If two different host entries on the same RADIUS server are configured for the same service--for example, authorization--the second host entry configured acts as fail-over backup to the first one. Using this example, if the first host entry fails to provide accounting services, the network access server will try the second host entry configured on the same device for accounting services. (The RADIUS host entries will be tried in the order they are configured.)

For more information about configuring server groups and about configuring server groups based on DNIS numbers, refer to the chapter Configuring RADIUS or the chapter Configuring TACACS+.

# AAA Authorization Types

Cisco IOS XE software supports five different types of authorization:

- Commands: Applies to the EXEC mode commands a user issues. Command authorization attempts authorization for all EXEC mode commands, including global configuration commands, associated with a specific privilege level.
- EXEC: Applies to the attributes associated with a user EXEC terminal session.
- Network: Applies to network connections. This can include a PPP, SLIP, or ARAP connection.
- Reverse Access: Applies to reverse Telnet sessions.
- Configuration: Applies to downloading configurations from the AAA server.
- IP Mobile: Applies to authorization for IP mobile services.

## Authorization Types

Named authorization method lists are specific to the indicated type of authorization.

To create a method list to enable authorization that applies specific security policies on a per-user basis, use the auth-proxy keyword. For detailed information on the authentication proxy feature, refer to the chapter "Configuring Authentication Proxy" in the "Traffic Filtering and Firewalls" part of this book.

To create a method list to enable authorization for all network-related service requests (including SLIP, PPP, PPP NCPs, and ARAP), use the **network** keyword.

To create a method list to enable authorization to determine if a user is allowed to run an EXEC shell, use the **exec** keyword.

To create a method list to enable authorization for specific, individual EXEC commands associated with a specific privilege level, use the **commands**keyword. (This allows you to authorize all commands associated with a specified command level from 0 to 15.)

To create a method list to enable authorization for reverse Telnet functions, use the **reverse-access** keyword.

For information about the types of authorization supported by the Cisco IOS XE software, refer to the AAA Authorization Types.

**Note** If a named server group used by the **aaa authorization** command is to be removed, make sure to remove the **aaa authorization** configuration also from the running configuration. If only the named server group is removed, and the **aaa authorization** command for that named server group is still present in the running-configuration, it can result in bulk sync failures and continuous reloads on HA devices.

If **aaa authorization** configurations are to be retained in the running configuration, make sure the associated named server group is configured with the correct settings.

## Authorization Attribute-Value Pairs

RADIUS and TACACS+ authorization both define specific rights for users by processing attributes, which are stored in a database on the security server. For both RADIUS and TACACS+, attributes are defined on the security server, associated with the user, and sent to the network access server where they are applied to the user's connection.

For a list of supported RADIUS attributes, refer to the "RADIUS Attributes Overview and RADIUS IETF Attributes" chapter. For a list of supported TACACS+ AV pairs, refer to the "Configuring TACACS+" chapter.

# How to Configure Authorization

## Configuring AAA Authorization Using Named Method Lists

To configure AAA authorization using named method lists, use the following commands beginning in global configuration mode:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** | Enables privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device> enable` | • Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa authorization** {**auth-proxy** \| **network** \| **exec** \| **commands** *level* \| **reverse-access** \| **configuration** \| **ipmobile**} {**default** \| *list-name*} [*method1* [*method2...*]]<br>**Example:**<br>`Device(config)# aaa authorization auth-proxy default` | Creates an authorization method list for a particular authorization type and enable authorization. |
| **Step 4** | Do one of the following:<br>• **line** [**aux** \| **console** \| **tty** \| **vty**] *line-number* [*ending-line-number*]<br>• **interface** *interface-type interface-number*<br>**Example:**<br>`Device(config)# line 1`<br>`Device(config)# interface gigabitethernet 0/1/1` | Enters the line configuration mode for the lines to which you want to apply the authorization method list.<br>Alternately, enters the interface configuration mode for the interfaces to which you want to apply the authorization method list. |
| **Step 5** | Do one of the following:<br>• **authorization**{**arap** \| **commands** *level* \| **exec** \| **reverse-access**} {**default** \| *list-name*}<br>• **ppp authorization**{**default** \| *list-name*}<br>**Example:**<br>`Device(config-line)# authorization commands default`<br>`Device(config-if)# ppp authorization default` | Applies the authorization list to a line or set of lines.<br>Alternately, applies the authorization list to an interface or set of interfaces. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-line)# end`<br>`Device(config-if)# end` | Exits line configuration mode and returns to privileged EXEC mode.<br>Exits interface configuration mode and returns to privileged EXEC mode. |

# Disabling Authorization for Global Configuration Commands

The **aaa authorization** command with the keyword **commands** attempts authorization for all EXEC mode commands, including global configuration commands, associated with a specific privilege level. Because there are configuration commands that are identical to some EXEC-level commands, there can be some confusion in the authorization process. Using **no aaa authorization config-commands** stops the network access server from attempting configuration command authorization.

When the **no aaa authorization console** command is configured along with the **aaa authentication enable default enable** command, enable mode will be locked unless enable password or enable secret is explicitly configured on the device.

To disable AAA authorization for all global configuration commands, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Device(config)# no aaa authorization config-commands` | Disables authorization for all global configuration commands. |

To disable AAA authorization on the console, use the following command in global configuration mode:

**Note** AAA authorization is disabled on the console by default. If AAA authorization is enabled on the console, disable it by configuring the **no aaa authorization console** command during the AAA configuration stage. AAA should be disabled on the console for user authentication.

| Command | Purpose |
|---|---|
| `Device(config)# no aaa authorization console` | Disables authorization on the console. |

# Configuring Authorization for Reverse Telnet

Telnet is a standard terminal emulation protocol used for remote terminal connection. Normally, you log in to a network access server (typically through a dialup connection) and then use Telnet to access other network devices from that network access server. There are times, however, when it is necessary to establish a reverse Telnet session. In reverse Telnet sessions, the Telnet connection is established in the opposite direction--from inside a network to a network access server on the network periphery to gain access to modems or other devices connected to that network access server. Reverse Telnet is used to provide users with dialout capability by allowing them to Telnet to modem ports attached to a network access server.

It is important to control access to ports accessible through reverse Telnet. Failure to do so could, for example, allow unauthorized users free access to modems where they can trap and divert incoming calls or make outgoing calls to unauthorized destinations.

Authentication during reverse Telnet is performed through the standard AAA login procedure for Telnet. Typically the user has to provide a username and password to establish either a Telnet or reverse Telnet session. Reverse Telnet authorization provides an additional (optional) level of security by requiring authorization in addition to authentication. When enabled, reverse Telnet authorization can use RADIUS or

TACACS+ to authorize whether or not this user is allowed reverse Telnet access to specific asynchronous ports, after the user successfully authenticates through the standard Telnet login procedure.

Reverse Telnet authorization offers the following benefits:

- An additional level of protection by ensuring that users engaged in reverse Telnet activities are indeed authorized to access a specific asynchronous port using reverse Telnet.
- An alternative method (other than access lists) to manage reverse Telnet authorization.

To configure a network access server to request authorization information from a TACACS+ or RADIUS server before allowing a user to establish a reverse Telnet session, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Device(config)# aaa authorization reverse-access method1 [method2 ...]` | Configures the network access server to request authorization information before allowing a user to establish a reverse Telnet session. |

This feature enables the network access server to request reverse Telnet authorization information from the security server, whether RADIUS or TACACS+. You must configure the specific reverse Telnet privileges for the user on the security server itself.

# Configuration Examples for Authorization

## Example: TACACS Authorization

The following examples show how to use a TACACS+ server to authorize the use of network services, including PPP and ARA. If the TACACS+ server is not available or an error occurs during the authorization process, the fallback method (none) is to grant all authorization requests:

```
Device(config)# aaa authorization network default group tacacs+ none
```

The following example shows how to allow network authorization using TACACS+:

```
Device(config)# aaa authorization network default group tacacs+
```

The following example shows how to provide the same authorization, but it also creates address pools called "*mci*" and "*att*":

```
Device> enable
Device# configure terminal
Device(config)# aaa authorization network default group tacacs+
Device(config)# interface gigabitethernet 01/1/
Device(config-if)# ip address-pool local
Device(config-if)# exit
Device(config)# ip local-pool mci 172.16.0.1 172.16.0.255
Device(config)# ip local-pool att 172.17.0.1 172.17.0.255
Device(config-if)# end
```

These address pools can then be selected by the TACACS daemon. A sample configuration of the daemon follows:

```
user = mci_customer1 {
    login = cleartext "some password"
    service = ppp protocol = ip {
        addr-pool=mci
    }
}
user = att_customer1 {
    login = cleartext "some other password"
    service = ppp protocol = ip {
        addr-pool=att
     }
```

# Example: RADIUS Authorization

The following example shows how to configure the router to authorize using RADIUS:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authorization exec default group radius if-authenticated
Device(config)# aaa authorization network default group radius
Device(config)# radius server ip
Device(config-radius-server)# key sharedkey
Device(config-radius-server)# end
```

The lines in this sample RADIUS authorization configuration are defined as follows:

- The **aaa authorization exec default group radius if-authenticated** command configures the network access server to contact the RADIUS server to determine if users are permitted to start an EXEC shell when they log in. If an error occurs when the network access server contacts the RADIUS server, the fallback method is to permit the CLI to start, provided the user has been properly authenticated.

The RADIUS information returned may be used to specify an autocommand or a connection access list be applied to this connection.

- The **aaa authorization network default group radius** command configures network authorization via RADIUS. This can be used to govern address assignment, the application of access lists, and various other per-user quantities.

**Note** Because no fallback method is specified in this example, authorization will fail if, for any reason, there is no response from the RADIUS server.

# Example: Reverse Telnet Authorization

The following examples show how to cause the network access server to request authorization information from a TACACS+ security server before allowing a user to establish a reverse Telnet session:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
```

```
Device(config)# aaa authentication login default group tacacs+
Device(config)# aaa authorization reverse-access default group tacacs+
Device(config)# tacacs server server1
Device(config-server-tacacs)# address ipv4 172.31.255.0
Device(config-server-tacacs)# timeout 90
Device(config-server-tacacs)# key sharedkey
Device(config-server-tacacs)# end
```

The lines in this sample TACACS+ reverse Telnet authorization configuration are defined as follows:

- The **aaa new-model** command enables AAA.
- The **aaa authentication login default group tacacs+** command specifies TACACS+ as the default method for user authentication during login.
- The **aaa authorization reverse-access default group tacacs+** command specifies TACACS+ as the method for user authorization when trying to establish a reverse Telnet session.
- The **tacacs server** command identifies the TACACS+ server.
- The **timeout** command sets the interval of time that the network access server waits for the TACACS+ server to reply.
- The **key** command defines the encryption key used for all TACACS+ communications between the network access server and the TACACS+ daemon.

The following example shows how to configure a generic TACACS+ server to grant a user, pat, reverse Telnet access to port tty2 on the network access server named “maple” and to port tty5 on the network access server named “oak”:

```
user = pat
  login = cleartext lab
  service = raccess {
    port#1 = maple/tty2
    port#2 = oak/tty5
```

**Note** In this example, “maple” and “oak” are the configured host names of network access servers, not DNS names or alias.

The following example shows how to configure the TACACS+ server (CiscoSecure) to grant a user named pat reverse Telnet access:

```
user = pat
profile_id = 90
profile_cycle = 1
member = Tacacs_Users
service=shell {
default cmd=permit
}
service=raccess {
allow “c2511e0” “tty1” “.*”
refuse “.*” “.*” “.*”
password = clear “goaway”
```

**Note** CiscoSecure only supports reverse Telnet using the command line interface in versions 2.1(*x*) through version 2.2(1).

An empty “service=raccess {}” clause permits a user to have unconditional access to network access server ports for reverse Telnet. If no “service=raccess” clause exists, the user is denied access to any port for reverse Telnet.

For more information about configuring TACACS+, refer to the “Configuring TACACS” chapter. For more information about configuring CiscoSecure, refer to the *CiscoSecure Access Control Server User Guide* , version 2.1(2) or greater.

The following example shows how to cause the network access server to request authorization from a RADIUS security server before allowing a user to establish a reverse Telnet session:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication login default group radius
Device(config)# aaa authorization reverse-access default group radius
Device(config)# radius server ip
Device(config-radius-server)# key sharedkey
Device(config-radius-server)# address ipv4 172.31.255.0 auth-port 1645 acct-port 1646
Device(config-radius-server)# end
```

The lines in this sample RADIUS reverse Telnet authorization configuration are defined as follows:

- The **aaa new-model** command enables AAA.
- The **aaa authentication login default group radius** command specifies RADIUS as the default method for user authentication during login.
- The **aaa authorization reverse-access default group radius** command specifies RADIUS as the method for user authorization when trying to establish a reverse Telnet session.
- The **radius** command identifies the RADIUS server.
- The **key** command defines the encryption key used for all RADIUS communications between the network access server and the RADIUS daemon.

The following example shows how to send a request to the RADIUS server to grant a user named “pat” reverse Telnet access at port tty2 on the network access server named “maple”:

```
Username = “pat”
Password = “goaway”
User-Service-Type = Shell-User
cisco-avpair = “raccess:port#1=maple/tty2”
```

The syntax "raccess:port=any/any" permits a user to have unconditional access to network access server ports for reverse Telnet. If no "raccess:port={*nasname* }/{*tty number* }" clause exists in the user profile, the user is denied access to reverse Telnet on all ports.

For more information about configuring RADIUS, refer to the chapter “Configuring RADIUS.”

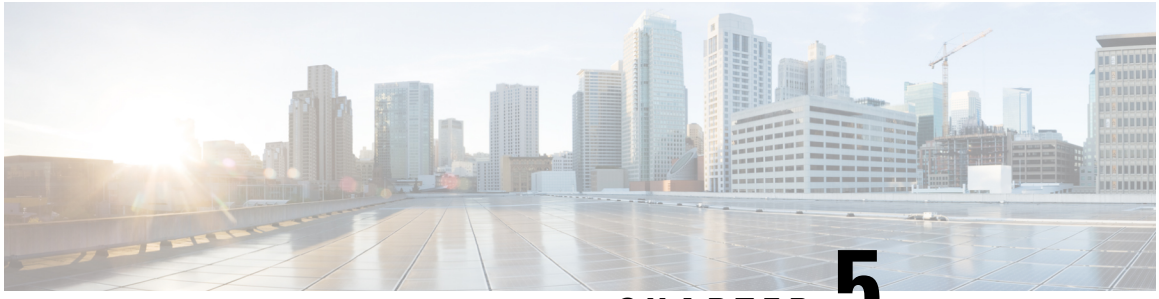
# Feature History for Configuring Authorization

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | AAA Authorization | AAA authorization enables you to limit the services available to a user. When AAA authorization is enabled, the network access server uses information retrieved from the user's profile, which is located either in the local user database or on the security server, to configure the user's session. Once this is done, the user will be granted access to a requested service only if the information in the user profile allows it.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | AAA Authorization | Support for this feature was introduced on the C9500-24Y4C, C9500-32C, C9500-32QC, and C9500-48Y4C modelsof the Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.



CHAPTER 5

# Configuring Accounting

The AAA accounting feature allows the services that users are accessing and the amount of network resources that users are consuming to be tracked. When AAA accounting is enabled, the network access server reports user activity to the TACACS+ or RADIUS security server (depending on which security method is implemented) in the form of accounting records. Each accounting record contains accounting attribute-value (AV) pairs and is stored on the security server. This data can then be analyzed for network management, client billing, and auditing.


- Prerequisites for Configuring Accounting, on page 89
- Restrictions for Configuring Accounting, on page 89
- Information About Configuring Accounting, on page 90
- How to Configure AAA Accounting, on page 103
- Configuration Examples for AAA Accounting, on page 110
- Additional References for Configuring Accounting, on page 114
- Feature History for Configuring Accounting, on page 115


## Prerequisites for Configuring Accounting

The following tasks must be performed before configuring accounting using named method lists:

- Enable AAA on the network access server by using the **aaa new-model**command in global configuration mode.
- Define the characteristics of the RADIUS or TACACS+ security server if RADIUS or TACACS+ authorization is issued. For more information about configuring the Cisco network access server to communicate with the RADIUS security server, see the Configuring RADIUS module. For more information about configuring the Cisco network access server to communicate with the TACACS+ security server, see the Configuring TACACS+ module.

## Restrictions for Configuring Accounting

- Accounting information can be sent simultaneously to a maximum of only four AAA servers.

# Information About Configuring Accounting

## Named Method Lists for Accounting

Similar to authentication and authorization method lists, method lists for accounting define the way accounting is performed and the sequence in which these methods are performed.

Named accounting method lists allow particular security protocol to be designated and used on specific lines or interfaces for accounting services. The only exception is the default method list (which is named default). The default method list is automatically applied to all interfaces except those that have a named method list explicitly defined. A defined method list overrides the default method list.

A method list is simply a named list describing the accounting methods to be queried (such as RADIUS or TACACS+), in sequence. Method lists allow one or more security protocols to be designated and used for accounting, thus ensuring a backup system for accounting in case the initial method fails. Cisco IOS software uses the first method listed to support accounting; if that method fails to respond, the Cisco IOS software selects the next accounting method listed in the method list. This process continues until there is successful communication with a listed accounting method, or all methods defined are exhausted.

**Note** The Cisco IOS software attempts accounting with the next listed accounting method only when there is no response from the previous method. If accounting fails at any point in this cycle--meaning that the security server responds by denying the user access--the accounting process stops and no other accounting methods are attempted.

Accounting method lists are specific to the type of accounting being requested. AAA supports seven different types of accounting:

- **Network**: Provides information for all PPP, SLIP, or ARAP sessions, including packet and byte counts.
- **EXEC**: Provides information about user EXEC terminal sessions of the network access server.
- **Commands**: Provides information about the EXEC mode commands that a user issues. Command accounting generates accounting records for all EXEC mode commands, including global configuration commands, associated with a specific privilege level.
- **Connection**: Provides information about all outbound connections made from the network access server, such as Telnet, local-area transport (LAT), TN3270, packet assembler/disassembler (PAD), and rlogin.
- **System**: Provides information about system-level events.
- **Resource**: Provides “start” and “stop” records for calls that have passed user authentication, and provides “stop” records for calls that fail to authenticate.
- **VRRS**: Provides information about Virtual Router Redundancy Service (VRRS).

**Note** System accounting does not use named accounting lists; only the default list for system accounting can be defined.

When a named method list is created, a particular list of accounting methods for the indicated accounting type are defined.

Accounting method lists must be applied to specific lines or interfaces before any of the defined methods are performed. The only exception is the default method list (which is named "default"). If the **aaa accounting** command for a particular accounting type is issued without specifying a named method list, the default method list is automatically applied to all interfaces or lines except those that have a named method list explicitly defined (A defined method list overrides the default method list). If no default method list is defined, then no accounting takes place.

**Note** If a named server group used by the **aaa accounting** command is to be removed, make sure to remove the **aaa accounting** configuration also from the running configuration. If only the named server group is removed, and the **aaa accounting** command for that named server group is still present in the running-configuration, it can result in bulk sync failures and continuous reloads on HA devices.

If **aaa accounting** configurations are to be retained in the running configuration, make sure the associated named server group is configured with the correct settings.

This section includes the following subsections:

## Method Lists and Server Groups

A server group is a way to group existing RADIUS or TACACS+ server hosts for use in method lists. The figure below shows a typical AAA network configuration that includes four security servers: R1 and R2 are RADIUS servers, and T1 and T2 are TACACS+ servers. R1 and R2 comprise the group of RADIUS servers. T1 and T2 comprise the group of TACACS+ servers.

In Cisco IOS software, RADIUS and TACACS+ server configurations are global. A subset of the configured server hosts can be specified using server groups. These server groups can be used for a particular service. For example, server groups allow R1 and R2 to be defined as separate server groups (SG1 and SG2), and T1 and T2 as separate server groups (SG3 and SG4). This means either R1 and T1 (SG1 and SG3) or R2 and T2 (SG2 and SG4) can be specified in the method list, which provides more flexibility in the way that RADIUS and TACACS+ resources are assigned.

Server groups also can include multiple host entries for the same server, as long as each entry has a unique identifier. The combination of an IP address and a UDP port number creates a unique identifier, allowing different ports to be individually defined as RADIUS hosts providing a specific AAA service. In other words, this unique identifier enables RADIUS requests to be sent to different UDP ports on a server from the same IP address. If two different host entries on the same RADIUS server are configured for the same service: for example, accounting; the second host entry configured acts as failover backup to the first one. Using this example, if the first host entry fails to provide accounting services, the network access server tries the second host entry configured on the same device for accounting services (The RADIUS host entries are tried in the order in which they are configured).

For more information about configuring server groups and about configuring server groups based on Dialed Number Identification Service (DNIS) numbers, see the "Configuring RADIUS" or "Configuring TACACS+" modules.

## AAA Accounting Methods

The following two methods of accounting are supported:

- TACACS+: The network access server reports user activity to the TACACS+ security server in the form of accounting records. Each accounting record contains accounting AV pairs and is stored on the security server.
- RADIUS: The network access server reports user activity to the RADIUS security server in the form of accounting records. Each accounting record contains accounting AV pairs and is stored on the security server.

**Note** Passwords and accounting logs are masked before being sent to the TACACS+ or RADIUS security servers. Use the **aaa accounting commands visible-keys** command to send unmasked information to the TACACS+ or RADIUS security servers.

## Accounting Record Types

For minimal accounting, use the **stop-only** keyword, which instructs the specified method (**RADIUS** or **TACACS+**) to send a stop record accounting notice at the end of the requested user process. For more accounting information, use the **start-stop** keyword to send a start accounting notice at the beginning of the requested event and a stop accounting notice at the end of the event. To stop all accounting activities on this line or interface, use the **none** keyword.

## Accounting Methods

The table below lists the supported accounting methods.

***Table 4: AAA Accounting Methods***

| Keyword | Description |
|---|---|
| **group radius** | Uses the list of all RADIUS servers for accounting. |
| **group tacacs+** | Uses the list of all TACACS+ servers for accounting. |
| **group** *group-name* | Uses a subset of RADIUS or TACACS+ servers for accounting as defined by the server group *group-name*. |

The method argument refers to the actual method the authentication algorithm tries. Additional methods of authentication are used only if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all other methods return an error, specify additional methods in the command. For example, to create a method list named acct_tac1 that specifies RADIUS as the backup method of authentication in the event that TACACS+ authentication returns an error, enter the following command:

```
aaa accounting network acct_tac1 stop-only group tacacs+ group radius
```

To create a default list that is used when a named list is not specified in the **aaa accounting**command, use the **default** keyword followed by the methods that are wanted to be used in default situations. The default method list is automatically applied to all interfaces.

For example, to specify RADIUS as the default method for user authentication during login, enter the following command:

```
aaa accounting network default stop-only group radius
```

AAA Accounting supports the following methods:

- **group tacacs** : To have the network access server send accounting information to a TACACS+ security server, use the **group tacacs+** *method* keyword.
- **group radius** : To have the network access server send accounting information to a RADIUS security server, use the **group radius** *method* keyword.

**Note** Accounting method lists for SLIP follow whatever is configured for PPP on the relevant interface. If no lists are defined and applied to a particular interface (or no PPP settings are configured), the default setting for accounting applies.

- **group** *group-name* : To specify a subset of RADIUS or TACACS+ servers to use as the accounting method, use the **aaa accounting**command with the **group** *group-name* method. To specify and define the group name and the members of the group, use the **aaa group server** command. For example, use the **aaa group server** command to first define the members of **group loginrad**:

```
aaa group server radius loginrad
 server 172.16.2.3
 server 172.16.2 17
 server 172.16.2.32
```

This command specifies RADIUS servers 172.16.2.3, 172.16.2.17, and 172.16.2.32 as members of the **group loginrad**.

To specify **group loginrad** as the method of network accounting when no other method list has been defined, enter the following command:

```
aaa accounting network default start-stop group loginrad
```

Before a group name can be used as the accounting method, communication with the RADIUS or TACACS+ security server must be enabled.

# AAA Accounting Types

## Network Accounting

Network accounting provides information for all PPP, SLIP, or ARAP sessions, including packet and byte counts.

The following example shows the information contained in a RADIUS network accounting record for a PPP user who comes in through an EXEC session:

```
Wed Jun 27 04:44:45 2001
        NAS-IP-Address = “172.16.25.15”
        NAS-Port = 5
        User-Name = “username1”
        Client-Port-DNIS = “4327528”
        Caller-ID = “562”
        Acct-Status-Type = Start
        Acct-Authentic = RADIUS
        Service-Type = Exec-User
```

```
        Acct-Session-Id = "0000000D"
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"

Wed Jun 27 04:45:00 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 5
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "562"
        Acct-Status-Type = Start
        Acct-Authentic = RADIUS
        Service-Type = Framed
        Acct-Session-Id = "0000000E"
        Framed-IP-Address = "10.1.1.2"
        Framed-Protocol = PPP
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
Wed Jun 27 04:47:46 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 5
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "562"
        Acct-Status-Type = Stop
        Acct-Authentic = RADIUS
        Service-Type = Framed
        Acct-Session-Id = "0000000E"
        Framed-IP-Address = "10.1.1.2"
        Framed-Protocol = PPP
        Acct-Input-Octets = 3075
        Acct-Output-Octets = 167
        Acct-Input-Packets = 39
        Acct-Output-Packets = 9
        Acct-Session-Time = 171
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
Wed Jun 27 04:48:45 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 5
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "408"
        Acct-Status-Type = Stop
        Acct-Authentic = RADIUS
        Service-Type = Exec-User
        Acct-Session-Id = "0000000D"
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
```

The following example shows the information contained in a TACACS+ network accounting record for a PPP user who first started an EXEC session:

```
Wed Jun 27 04:00:35 2001 172.16.25.15    username1   tty4    562/4327528     starttask_id=28
      service=shell
Wed Jun 27 04:00:46 2001 172.16.25.15    username1   tty4    562/4327528     starttask_id=30
      addr=10.1.1.1   service=ppp
Wed Jun 27 04:00:49 2001 172.16.25.15    username1   tty4    408/4327528     updattask_id=30
      addr=10.1.1.1   service=ppp     protocol=ip     addr=10.1.1.1
Wed Jun 27 04:01:31 2001 172.16.25.15    username1   tty4    562/4327528     stoptask_id=30
```

```
       addr=10.1.1.1   service=ppp     protocol=ip     addr=10.1.1.1
                                         bytes_in=2844        bytes_out=1682  paks_in=36
    paks_out=24     elapsed_time=51
Wed Jun 27 04:01:32 2001 172.16.25.15    username1   tty4    562/4327528     stoptask_id=28
       service=shell   elapsed_time=57
```

**Note** The precise format of accounting packets records may vary depending on the security server daemon.

The following example shows the information contained in a RADIUS network accounting record for a PPP user who comes in through autoselect:

```
Wed Jun 27 04:30:52 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 3
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "562"
        Acct-Status-Type = Start
        Acct-Authentic = RADIUS
        Service-Type = Framed
        Acct-Session-Id = "0000000B"
        Framed-Protocol = PPP
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"

Wed Jun 27 04:36:49 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 3
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "562"
        Acct-Status-Type = Stop
        Acct-Authentic = RADIUS
        Service-Type = Framed
        Acct-Session-Id = "0000000B"
        Framed-Protocol = PPP
        Framed-IP-Address = "10.1.1.1"
        Acct-Input-Octets = 8630
        Acct-Output-Octets = 5722
        Acct-Input-Packets = 94
        Acct-Output-Packets = 64
        Acct-Session-Time = 357
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
```

The following example shows the information contained in a TACACS+ network accounting record for a PPP user who comes in through autoselect:

```
Wed Jun 27 04:02:19 2001 172.16.25.15    username1   Async5  562/4327528     starttask_id=35
      service=ppp
Wed Jun 27 04:02:25 2001 172.16.25.15    username1   Async5  562/4327528     updatetask_id=35
     service=ppp     protocol=ip     addr=10.1.1.2
Wed Jun 27 04:05:03 2001 172.16.25.15    username1   Async5  562/4327528     stoptask_id=35
       service=ppp     protocol=ip     addr=10.1.1.2
                                         bytes_in=3366   bytes_out=2149       paks_in=42
    paks_out=28     elapsed_time=164
```

# EXEC Accounting

EXEC accounting provides information about user EXEC terminal sessions (user shells) on the network access server, including username, date, start and stop times, the access server IP address, and (for dial-in users) the telephone number the call originated from.

The following example shows the information contained in a RADIUS EXEC accounting record for a dial-in user:

```
Wed Jun 27 04:26:23 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 1
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329483"
        Acct-Status-Type = Start
        Acct-Authentic = RADIUS
        Service-Type = Exec-User
        Acct-Session-Id = "00000006"
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
Wed Jun 27 04:27:25 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 1
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329483"
        Acct-Status-Type = Stop
        Acct-Authentic = RADIUS
        Service-Type = Exec-User
        Acct-Session-Id = "00000006"
        Acct-Session-Time = 62
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
```

The following example shows the information contained in a TACACS+ EXEC accounting record for a dial-in user:

```
Wed Jun 27 03:46:21 2001        172.16.25.15    username1   tty3    5622329430/4327528
start
 task_id=2       service=shell
Wed Jun 27 04:08:55 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop
 task_id=2       service=shell   elapsed_time=1354
```

The following example shows the information contained in a RADIUS EXEC accounting record for a Telnet user:

```
Wed Jun 27 04:48:32 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 26
        User-Name = "username1"
        Caller-ID = "10.68.202.158"
        Acct-Status-Type = Start
        Acct-Authentic = RADIUS
        Service-Type = Exec-User
        Acct-Session-Id = "00000010"
        Acct-Delay-Time = 0
        User-Id = "username1"
```

```
        NAS-Identifier = "172.16.25.15"

Wed Jun 27 04:48:46 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 26
        User-Name = "username1"
        Caller-ID = "10.68.202.158"
        Acct-Status-Type = Stop
        Acct-Authentic = RADIUS
        Service-Type = Exec-User
        Acct-Session-Id = "00000010"
        Acct-Session-Time = 14
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
```

The following example shows the information contained in a TACACS+ EXEC accounting record for a Telnet user:

```
Wed Jun 27 04:06:53 2001        172.16.25.15    username1   tty26   10.68.202.158
starttask_id=41      service=shell
Wed Jun 27 04:07:02 2001        172.16.25.15    username1   tty26   10.68.202.158
stoptask_id=41       service=shell   elapsed_time=9
```

## Command Accounting

Command accounting provides information about the EXEC shell commands for a specified privilege level that are being executed on a network access server. Each command accounting record includes a list of the commands executed for that privilege level, as well as the date and time each command was executed, and the user who executed it.

The following example shows the information contained in a TACACS+ command accounting record for privilege level 1:

```
Wed Jun 27 03:46:47 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=3       service=shell   priv-lvl=1      cmd=show version <cr>
Wed Jun 27 03:46:58 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=4       service=shell   priv-lvl=1      cmd=show interfaces Ethernet 0
<cr>
Wed Jun 27 03:47:03 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=5       service=shell   priv-lvl=1      cmd=show ip route <cr>
```

The following example shows the information contained in a TACACS+ command accounting record for privilege level 15:

```
Wed Jun 27 03:47:17 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=6       service=shell   priv-lvl=15     cmd=configure terminal <cr>
Wed Jun 27 03:47:21 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=7       service=shell   priv-lvl=15     cmd=interface Serial 0 <cr>
Wed Jun 27 03:47:29 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop    task_id=8      service=shell   priv-lvl=15    cmd=ip address 10.1.1.1 255.255.255.0
 <cr>
```

**Note** The Cisco implementation of RADIUS does not support command accounting.

## Connection Accounting

Connection accounting provides information about all outbound connections made from the network access server such as Telnet, LAT, TN3270, PAD, and rlogin.

The following example shows the information contained in a RADIUS connection accounting record for an outbound Telnet connection:

```
Wed Jun 27 04:28:00 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 2
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329477"
        Acct-Status-Type = Start
        Acct-Authentic = RADIUS
        Service-Type = Login
        Acct-Session-Id = "00000008"
        Login-Service = Telnet
        Login-IP-Host = "10.68.202.158"
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"

Wed Jun 27 04:28:39 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 2
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329477"
        Acct-Status-Type = Stop
        Acct-Authentic = RADIUS
        Service-Type = Login
        Acct-Session-Id = "00000008"
        Login-Service = Telnet
        Login-IP-Host = "10.68.202.158"
        Acct-Input-Octets = 10774
        Acct-Output-Octets = 112
        Acct-Input-Packets = 91
        Acct-Output-Packets = 99
        Acct-Session-Time = 39
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
```

The following example shows the information contained in a TACACS+ connection accounting record for an outbound Telnet connection:

```
Wed Jun 27 03:47:43 2001        172.16.25.15    username1   tty3    5622329430/4327528
start    task_id=10     service=connection     protocol=telnet addr=10.68.202.158 cmd=telnet
 username1-sun
Wed Jun 27 03:48:38 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=10     service=connection     protocol=telnet addr=10.68.202.158 cmd=telnet
 username1-sun    bytes_in=4467   bytes_out=96    paks_in=61     paks_out=72 elapsed_time=55
```

The following example shows the information contained in a RADIUS connection accounting record for an outbound rlogin connection:

```
Wed Jun 27 04:29:48 2001
        NAS-IP-Address = "172.16.25.15"
```

```
        NAS-Port = 2
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329477"
        Acct-Status-Type = Start
        Acct-Authentic = RADIUS
        Service-Type = Login
        Acct-Session-Id = "0000000A"
        Login-Service = Rlogin
        Login-IP-Host = "10.68.202.158"
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"

Wed Jun 27 04:30:09 2001
        NAS-IP-Address = "172.16.25.15"
        NAS-Port = 2
        User-Name = "username1"
        Client-Port-DNIS = "4327528"
        Caller-ID = "5622329477"
        Acct-Status-Type = Stop
        Acct-Authentic = RADIUS
        Service-Type = Login
        Acct-Session-Id = "0000000A"
        Login-Service = Rlogin
        Login-IP-Host = "10.68.202.158"
        Acct-Input-Octets = 18686
        Acct-Output-Octets = 86
        Acct-Input-Packets = 90
        Acct-Output-Packets = 68
        Acct-Session-Time = 22
        Acct-Delay-Time = 0
        User-Id = "username1"
        NAS-Identifier = "172.16.25.15"
```

The following example shows the information contained in a TACACS+ connection accounting record for an outbound rlogin connection:

```
Wed Jun 27 03:48:46 2001        172.16.25.15    username1   tty3    5622329430/4327528
start    task_id=12     service=connection     protocol=rlogin addr=10.68.202.158 cmd=rlogin
 username1-sun /user username1
Wed Jun 27 03:51:37 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=12     service=connection     protocol=rlogin addr=10.68.202.158 cmd=rlogin
 username1-sun /user username1 bytes_in=659926 bytes_out=138   paks_in=2378    paks_
out=1251        elapsed_time=171
```

The following example shows the information contained in a TACACS+ connection accounting record for an outbound LAT connection:

```
Wed Jun 27 03:53:06 2001        172.16.25.15    username1   tty3    5622329430/4327528
start    task_id=18      service=connection      protocol=lat    addr=VAX        cmd=lat
VAX
Wed Jun 27 03:54:15 2001        172.16.25.15    username1   tty3    5622329430/4327528
stop     task_id=18      service=connection      protocol=lat    addr=VAX        cmd=lat
VAX  bytes_in=0      bytes_out=0     paks_in=0      paks_out=0      elapsed_time=6
```

## System Accounting

System accounting provides information about all system-level events (for example, when the system reboots or when accounting is turned on or off).

The following accounting record shows a typical TACACS+ system accounting record server indicating that AAA Accounting has been turned off:

```
Wed Jun 27 03:55:32 2001        172.16.25.15    unknown unknown unknown start   task_id=25
   service=system
event=sys_acct  reason=reconfigure
```

**Note** The precise format of accounting packets records may vary depending on the TACACS+ daemon.

The following accounting record shows a TACACS+ system accounting record indicating that AAA Accounting has been turned on:

```
Wed Jun 27 03:55:22 2001        172.16.25.15    unknown unknown unknown stop    task_id=23
   service=system
event=sys_acct  reason=reconfigure
```

## Resource Accounting

The Cisco implementation of AAA accounting provides “start” and “stop” record support for calls that have passed user authentication. The additional feature of generating “stop” records for calls that fail to authenticate as part of user authentication is also supported. Such records are necessary for users employing accounting records to manage and monitor their networks.

This section includes the following subsections:

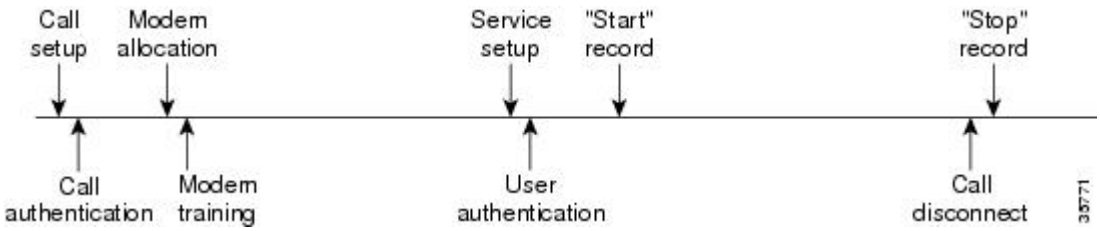
### AAA Resource Failure Stop Accounting

Before AAA resource failure stop accounting, there was no method of providing accounting records for calls that failed to reach the user authentication stage of a call setup sequence. Such records are necessary for users employing accounting records to manage and monitor their networks and their wholesale customers.

This functionality generates a “stop” accounting record for any calls that do not reach user authentication; “stop” records are generated from the moment of call setup. All calls that pass user authentication behave as they did before; that is, no additional accounting records are seen.

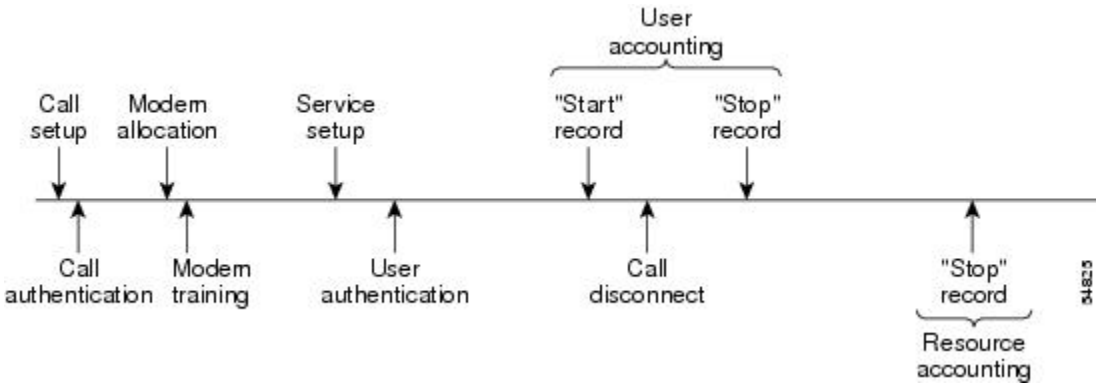
The figure below illustrates a call setup sequence with normal call flow (no disconnect) and without AAA resource failure stop accounting enabled.

***Figure 4: Modem Dial-In Call Setup Sequence With Normal Flow and Without Resource Failure Stop Accounting Enabled***




The figure below illustrates a call setup sequence with normal call flow (no disconnect) and with AAA resource failure stop accounting enabled.

*Figure 5: Modem Dial-In Call Setup Sequence With Normal Flow and WIth Resource Failure Stop Accounting Enabled*

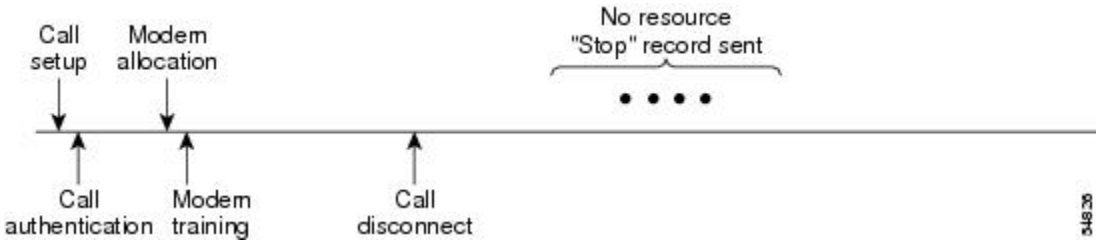



The figure below illustrates a call setup sequence with call disconnect occurring before user authentication and with AAA resource failure stop accounting enabled.

*Figure 6: Modem Dial-In Call Setup Sequence With Call Disconnect Occurring Before User Authentication and With Resource Failure Stop Accounting Enabled*

The figure below illustrates a call setup sequence with call disconnect occurring before user authentication and without AAA resource failure stop accounting enabled.

*Figure 7: Modem Dial-In Call Setup Sequence With Call Disconnect Occurring Before User Authentication and Without Resource Failure Stop Accounting Enabled*



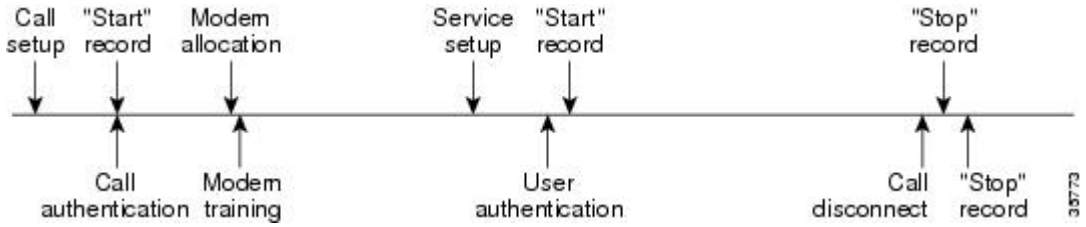

## AAA Resource Accounting for Start-Stop Records

AAA resource accounting for start-stop records supports the ability to send a “start” record at each call setup, followed by a corresponding “stop” record at the call disconnect. This functionality can be used to manage and monitor wholesale customers from one source of data reporting, such as accounting records.

With this feature, a call setup and call disconnect “start-stop” accounting record tracks the progress of the resource connection to the device. A separate user authentication “start-stop” accounting record tracks the user management progress. These two sets of accounting records are interlinked by using a unique session ID for the call.

The figure below illustrates a call setup sequence with AAA resource start-stop accounting enabled.

*Figure 8: Modem Dial-In Call Setup Sequence With Resource Start-Stop Accounting Enabled*




# AAA Accounting Enhancements

## AAA Broadcast Accounting

AAA broadcast accounting allows accounting information to be sent to multiple AAA servers at the same time; that is, accounting information can be broadcast to one or more AAA servers simultaneously. This functionality allows service providers to send accounting information to their own private AAA servers and to the AAA servers of their end customers. It also provides redundant billing information for voice applications.

Broadcasting is allowed among groups of RADIUS or TACACS+ servers, and each server group can define its backup servers for failover independently of other groups.

Thus, service providers and their end customers can use different protocols (RADIUS or TACACS+) for the accounting server. Service providers and their end customers can also specify their backup servers independently. As for voice applications, redundant accounting information can be managed independently through a separate group with its own failover sequence.

## AAA Session MIB

The AAA session MIB feature allows customers to monitor and terminate their authenticated client connections using Simple Network Management Protocol (SNMP). The data of the client is presented so that it correlates directly to the AAA Accounting information reported by either the RADIUS or the TACACS+ server. AAA session MIB provides the following information:

- Statistics for each AAA function (when used in conjunction with the **show radius statistics** command)
- Status of servers providing AAA functions
- Identities of external AAA servers
- Real-time information (such as idle times), providing additional criteria for use by SNMP networks for assessing whether or not to terminate an active call

The table below shows the SNMP user-end data objects that can be used to monitor and terminate authenticated client connections with the AAA session MIB feature.

*Table 5: SNMP End-User Data Objects*

| | |
|---|---|
| SessionId | The session identification used by the AAA Accounting protocol (same value as reported by RADIUS attribute 44 (Acct-Session-ID)). |
| UserId | The user login ID or zero-length string if a login is unavailable. |
| IpAddr | The IP address of the session or 0.0.0.0 if an IP address is not applicable or unavailable. |

| | |
|---|---|
| IdleTime | The elapsed time in seconds that the session has been idle. |
| Disconnect | The session termination object used to disconnect the given client. |
| CallId | The entry index corresponding to this accounting session that the Call Tracker record stored. |

The table below describes the AAA summary information provided by the AAA session MIB feature using SNMP on a per-system basis.

***Table 6: SNMP AAA Session Summary***

| | |
|---|---|
| ActiveTableEntries | Number of sessions currently active. |
| ActiveTableHighWaterMark | Maximum number of sessions present at once since last system reinstallation. |
| TotalSessions | Total number of sessions since last system reinstallation. |
| DisconnectedSessions | Total number of sessions that have been disconnected using since last system reinstallation. |

## Accounting Attribute-Value Pairs

The network access server monitors the accounting functions defined in either TACACS+ AV pairs or RADIUS attributes, depending on which security method is implemented.

# How to Configure AAA Accounting

## Configuring AAA Accounting Using Named Method Lists

To configure AAA Accounting using named method lists, perform the following steps:

**Note** System accounting does not use named method lists. For system accounting, define only the default method list.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | **aaa accounting** {**system** \| **network** \| **exec** \| **connection** \| **commands** *level*} {**default** \| *list-name*} {**start-stop** \| **stop-only** \| **none**} [*method1* [*method2...*]]<br>**Example:**<br>`Device(config)# aaa accounting system default start-stop` | Creates an accounting method list and enables accounting. The argument *list-name* is a character string used to name the created list. |
| **Step 4** | Do one of the following:<br>• **line** [**aux** \| **console** \| **tty** \| **vty**] *line-number* [*ending-line-number*]<br>• **interface** *interface-type interface-number*<br>**Example:**<br>`Device(config)# line aux line1` | Enters the line configuration mode for the lines to which the accounting method list is applied.<br>or<br>Enters the interface configuration mode for the interfaces to which the accounting method list is applied. |
| **Step 5** | Do one of the following:<br>• **accounting** {**arap** \| **commands** *level* \| **connection** \| **exec**} {**default** \| *list-name*}<br>• **ppp accounting**{**default** \| *list-name*}<br>**Example:**<br>`Device(config-line)# accounting arap default` | Applies the accounting method list to a line or set of lines.<br>or<br>Applies the accounting method list to an interface or set of interfaces. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-line)# end` | (Optional) Exits line configuration mode and returns to privileged EXEC mode. |

# Suppressing Generation of Accounting Records for Null Username Sessions

When AAA Accounting is activated, the Cisco IOS software issues accounting records for all users on the system, including users whose username string, because of protocol translation, is NULL. An example of this is users who come in on lines where the **aaa authentication login** *method-list* **none** command is applied. To prevent accounting records from being generated for sessions that do not have usernames associated with them, use the following command in global configuration mode:

| Command | Purpose |
| --- | --- |
| `Device(config)# aaa accounting suppress null-username` | Prevents accounting records from being generated for users whose username string is NULL. |

# Generating Interim Accounting Records

To enable periodic interim accounting records to be sent to the accounting server, use the following command in global configuration mode:

| Command | Purpose |
| --- | --- |
| `Device(config)# aaa accounting update [newinfo] [periodic] number` | Enables periodic interim accounting records to be sent to the accounting server. |

When the **aaa accounting update**commandis activated, the Cisco IOS software issues interim accounting records for all users on the system. If the keyword **newinfo** is used, interim accounting records are sent to the accounting server every time there is new accounting information to report. An example of this would be when IPCP completes IP address negotiation with the remote peer. The interim accounting record includes the negotiated IP address used by the remote peer.

When used with the keyword **periodic**, interim accounting records are sent periodically as defined by the *number* argument. The interim accounting record contains all of the accounting information recorded for that user up to the time the interim accounting record is sent.

**Caution** Using the **aaa accounting update periodic** command can cause heavy congestion when many users are logged in to the network.

# Configuring an Alternate Method to Enable Periodic Accounting Records

You can use the following alternative method to enable periodic interim accounting records to be sent to the accounting server.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa accounting network default**<br>**Example:**<br>`Device(config)# aaa accounting network default` | Configures the default accounting for all network-related service requests and enters accounting method list configuration mode. |
| **Step 4** | **action-type** {**none** \| **start-stop** [**periodic** {**disable** \| **interval** *minutes*}] \| **stop-only**}<br>**Example:**<br>`Device(cfg-acct-mlist)# action-type start-stop`<br>**Example:**<br>`periodic interval 5` | Specifies the type of action to be performed on accounting records.<br>• (Optional) The periodic keyword specifies periodic accounting action.<br>• The interval keyword specifies the periodic accounting interval.<br>• The *value* argument specifies the intervals for accounting update records (in minutes).<br>• The **disable** keyword disables periodic accounting. |
| **Step 5** | **end**<br>**Example:**<br>`Device(cfg-acct-mlist)# end` | Exits accounting method list configuration mode and returns to privileged EXEC mode. |

# Generating Interim Service Accounting Records

Perform this task to enable the generation of interim service accounting records at periodic intervals for subscribers.

### Before you begin

RADIUS Attribute 85 in the user service profile always takes precedence over the configured interim-interval value. RADIUS Attribute 85 must be in the user service profile. See the RADIUS Attributes Overview and RADIUS IETF Attributes feature document for more information.

**Note** If RADIUS Attribute 85 is not in the user service profile, then the interim-interval value configured in Generating Interim Accounting Records is used for service interim accounting records.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **subscriber service accounting interim-interval** *minutes*<br>**Example:**<br>`Device(config)# subscriber service accounting interim-interval 10` | Enables the generation of interim service accounting records at periodic intervals for subscribers. The *minutes* argument indicates the number of periodic intervals to send accounting update records from 1 to 71582 minutes. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Generating Accounting Records for a Failed Login or Session

When AAA accounting is activated, the Cisco IOS XE software does not generate accounting records for system users who fail login authentication, or who succeed in login authentication but fail PPP negotiation for some reason.

To specify that accounting stop records be generated for users who fail to authenticate at login or during session negotiation, use the following command in global configuration mode:

| Command or Action | Purpose |
|---|---|
| **aaa accounting send stop-record authentication failure** | Generates “stop” records for users who fail to authenticate at login or during session negotiation using PPP. |

# Specifying Accounting NETWORK-Stop Records Before EXEC-Stop Records

For PPP users who start EXEC terminal sessions, it can be specified that NETWORK records be generated before EXEC-stop records. In some cases, such as billing customers for specific services, is can be desirable to keep network start and stop records together, essentially "nesting" them within the framework of the EXEC start and stop messages. For example, a user dialing in using PPP can create the following records: EXEC-start, NETWORK-start, EXEC-stop, NETWORK-stop. By nesting the network accounting records, NETWORK-stop records follow NETWORK-start messages: EXEC-start, NETWORK-start, NETWORK-stop, EXEC-stop.

To nest accounting records for user sessions, use the following command in global configuration mode:

| Command or Action | Purpose |
|---|---|
| **aaa accounting nested** | Nests network accounting records. |

# Suppressing System Accounting Records over Switchover

To suppress the system accounting-on and accounting-off messages during switchover, use the following command in global configuration mode:

| Command or Action | Purpose |
|---|---|
| **aaa accounting redundancy suppress system-records** | Suppresses the system accounting messages during switchover. |

# Configuring AAA Resource Failure Stop Accounting

To enable resource failure stop accounting, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Device(config)# aaa accounting resource` *method-list* `stop-failure group` *server-group* | Generates a "stop" record for any calls that do not reach user authentication.<br>**Note**<br>Before configuring this feature, the tasks described in the Prerequisites for Configuring Accounting, on page 89 section must be performed, and SNMP must be enabled on the network access server. |

# Configuring AAA Resource Accounting for Start-Stop Records

To enable full resource accounting for start-stop records, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Device(config)# aaa accounting resource method-list start-stop group server-group` | Supports the ability to send a "start" record at each call setup. followed with a corresponding "stop" record at the call disconnect.<br>**Note**<br>Before configuring this feature, the tasks described in the Prerequisites for Configuring Accounting, on page 89 section must be performed, and SNMP must be enabled on the network access server. |

# AAA Broadcast Accounting

AAA broadcast accounting allows accounting information to be sent to multiple AAA servers at the same time; that is, accounting information can be broadcast to one or more AAA servers simultaneously. This functionality allows service providers to send accounting information to their own private AAA servers and to the AAA servers of their end customers. It also provides redundant billing information for voice applications.

Broadcasting is allowed among groups of RADIUS or TACACS+ servers, and each server group can define its backup servers for failover independently of other groups.

Thus, service providers and their end customers can use different protocols (RADIUS or TACACS+) for the accounting server. Service providers and their end customers can also specify their backup servers independently. As for voice applications, redundant accounting information can be managed independently through a separate group with its own failover sequence.

# Configuring Per-DNIS AAA Broadcast Accounting

To configure AAA broadcast accounting per DNIS, use the **aaa dnis map accounting network**command in global configuration mode:

| Command | Purpose |
|---|---|
| `Device(config)# aaa dnis map dnis-number accounting network [start-stop \| stop-only \| none] [broadcast] method1 [method2...]` | Allows per-DNIS accounting configuration. This command has precedence over the global **aaa accounting** command.<br>Enables sending accounting records to multiple AAA servers. Simultaneously sends accounting records to the first server in each group. If the first server is unavailable, failover occurs using the backup servers defined within that group. |

# Establishing a Session with a Device if the AAA Server Is Unreachable

To establish a console session with a device if the AAA server is unreachable, use the following command in global configuration mode:

| Command or Action | Purpose |
|---|---|
| **no aaa accounting system guarantee-first** | The **aaa accounting system guarantee-first** command guarantees system accounting as the first record, which is the default condition.<br><br>In some situations, users may be prevented from starting a session on the console or terminal connection until after the system reloads, which can take more than three minutes. To resolve this problem, use the **no aaa accounting system guarantee-first** command. |

## Monitoring Accounting

No specific **show** command exists for either RADIUS or TACACS+ accounting. To obtain accounting records displaying information about users logged in, use the following command in privileged EXEC mode:

| Command or Action | Purpose |
|---|---|
| **show accounting** | Allows display of the active accountable events on the network and helps collect information in the event of a data loss on the accounting server. |

## Troubleshooting Accounting

To troubleshoot accounting information, use the following command in privileged EXEC mode:

| Command or Action | Purpose |
|---|---|
| **debug aaa accounting** | Displays information on accountable events as they occur. |

# Configuration Examples for AAA Accounting

## Example: Configuring a Named Method List

The following example shows how to configure a Cisco device (enabled for AAA and communication with a RADIUS security server) in order for AAA services to be provided by the RADIUS server. If the RADIUS server fails to respond, then the local database is queried for authentication and authorization information, and accounting services are handled by a TACACS+ server.

```
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication login admins local
Device(config)# aaa authentication ppp dialins group radius local
Device(config)# aaa authorization network network1 group radius local
Device(config)# aaa accounting network network2 start-stop group radius group tacacs+
Device(config)# username root password ALongPassword
Device(config)# tacacs server server1
Device(config-server-tacacs)# address ipv4 172.31.255.0
Device(config-server-tacacs)# key goaway
Device(config-server-tacacs)# exit
Device(config)# radius server isp
Device(config-sg-radius)# key myRaDiUSpassWoRd
```

```
Device(config-sg-radius)# exit
Device(config)# interface group-async 1
Device(config-if)# group-range 1 16
Device(config-if)# encapsulation ppp
Device(config-if)# ppp authentication chap dialins
Device(config-if)# ppp authorization network1
Device(config-if)# ppp accounting network2
Device(config-if)# exit
Device(config)# line 1 16
Device(config-line)# autoselect ppp
Device(config-line)# autoselect during-login
Device(config-line)# login authentication admins
Device(config-line)# modem dialin
Device(config-line)# end
```

The lines in this sample RADIUS AAA configuration are defined as follows:

- The **aaa new-model** command enables AAA network security services.
- The **aaa authentication login admins local** command defines a method list, “admins”, for login authentication.
- The **aaa authentication ppp dialins group radius local** command defines the authentication method list “dialins”, which specifies that first RADIUS authentication and then (if the RADIUS server does not respond) local authentication is used on serial lines using PPP.
- The **aaa authorization network network1 group radius local** command defines the network authorization method list named “network1”, which specifies that RADIUS authorization is used on serial lines using PPP. If the RADIUS server fails to respond, then local network authorization is performed.
- The **aaa accounting network network2 start-stop group radius group tacacs+** command defines the network accounting method list named “network2”, which specifies that RADIUS accounting services (in this case, start and stop records for specific events) are used on serial lines using PPP. If the RADIUS server fails to respond, accounting services are handled by a TACACS+ server.
- The **username** command defines the username and password to be used for the PPP Password Authentication Protocol (PAP) caller identification.
- The **tacacs server** command defines the name of the TACACS+ server host.
- The **key** command defines the shared secret text string between the network access server and the TACACS+ server host.
- The **radius server** command defines the name of the RADIUS server host.
- The **key** command defines the shared secret text string between the network access server and the RADIUS server host.
- The **interface group-async** command selects and defines an asynchronous interface group.
- The **group-range** command defines the member asynchronous interfaces in the interface group.
- The**encapsulation ppp** command sets PPP as the encapsulation method used on the specified interfaces.
- The**ppp authentication chap dialins** command selects Challenge Handshake Authentication Protocol (CHAP) as the method of PPP authentication and applies the “dialins” method list to the specified interfaces.

- The**ppp authorization network1** command applies the blue1 network authorization method list to the specified interfaces.
- The**ppp accounting network2** command applies the red1 network accounting method list to the specified interfaces.
- The **line** command switches the configuration mode from global configuration to line configuration and identifies the specific lines being configured.
- The **autoselect ppp** command configures the Cisco IOS XE software to allow a PPP session to start up automatically on these selected lines.
- The **autoselect during-login** command is used to display the username and password prompt without pressing the Return key. After the user logs in, the autoselect function (in this case, PPP) begins.
- The **login authentication admins** command applies the admins method list for login authentication.
- The **modem dialin** command configures modems attached to the selected lines to accept only incoming calls.

The **show accounting** command yields the following output for the preceding configuration:

```
Active Accounted actions on tty1, User username2 Priv 1
 Task ID 5, Network Accounting record, 00:00:52 Elapsed
 task_id=5 service=ppp protocol=ip address=10.0.0.98
```

The table below describes the fields contained in the preceding output.

***Table 7: show accounting Field Descriptions***

| Field | Description |
|---|---|
| Active Accounted actions on | Terminal line or interface name user with which the user logged in. |
| User | User's ID. |
| Priv | User's privilege level. |
| Task ID | Unique identifier for each accounting session. |
| Accounting Record | Type of accounting session. |
| Elapsed | Length of time (hh:mm:ss) for this session type. |
| attribute=value | AV pairs associated with this accounting session. |

# Example: Configuring AAA Resource Accounting

The following example shows how to configure the resource failure stop accounting and resource accounting for start-stop records functions:

```
!Enable AAA on your network access server.
Device(config)# aaa new-model
!Enable authentication at login and list the AOL string name to use for login authentication.
Device(config)# aaa authentication login AOL group radius local
!Enable authentication for ppp and list the default method to use for PPP authentication.
Device(config)# aaa authentication ppp default group radius local
```

```
!Enable authorization for all exec sessions and list the AOL string name to use for
authorization.
Device(config)# aaa authorization exec AOL group radius if-authenticated
!Enable authorization for all network-related service requests and list the default method

 to use for all network-related authorizations.
Device(config)# aaa authorization network default group radius if-authenticated
!Enable accounting for all exec sessions and list the default method to use for all start-stop

  accounting services.
Device(config)# aaa accounting exec default start-stop group radius
!Enable accounting for all network-related service requests and list the default method to
 use
 for all start-stop accounting services.
Device(config)# aaa accounting network default start-stop group radius
!Enable failure stop accounting.
Device(config)# aaa accounting resource default stop-failure group radius
!Enable resource accounting for start-stop records.
Device(config)# aaa accounting resource default start-stop group radius
```

# Example: Configuring AAA Broadcast Accounting

The following example shows how to turn on broadcast accounting using the global **aaa accounting** command:

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius isp
Device(config-sg-radius)# server 10.0.0.1
Device(config-sg-radius)# server 10.0.0.2
Device(config-sg-radius)# exit
Device(config)# aaa group server tacacs+ isp_customer
Device config-sg-tacacs+)# server 172.0.0.1
Device config-sg-tacacs+)# exit
Device(config)# aaa accounting network default start-stop broadcast group isp group
isp_customer
Device(config)# tacacs server server1
Device(config-server-tacacs)# address ipv4 172.0.0.1
Device(config-server-tacacs)# key key2
Device(config-server-tacacs)# end
```

The **broadcast** keyword causes “start” and “stop” accounting records for network connections to be sent simultaneously to server 10.0.0.1 in the group isp and to server 172.0.0.1 in the group isp_customer. If server 10.0.0.1 is unavailable, failover to server 10.0.0.2 occurs. If server 172.0.0.1 is unavailable, no failover occurs because backup servers are not configured for the group isp_customer.

# Example: Configuring per-DNIS AAA Broadcast Accounting

The following example shows how to turn on per-DNIS broadcast accounting using the global **aaa dnis map accounting network** command:

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius isp
Device(config-sg-radius)# server 10.0.0.1
Device(config-sg-radius)# server 10.0.0.2
Device(config-sg-radius)# exit
Device(config)# aaa group server tacacs+ isp_customer
Device config-sg-tacacs+)# server 172.0.0.1
Device config-sg-tacacs+)# exit
Device(config)# aaa dnis map enable
```

```
Device(config)# aaa dnis map 7777 accounting network start-stop broadcast group isp group
isp_customer
Device(config)# tacacs server server1
Device(config-server-tacacs)# address ipv4 172.0.0.1
Device(config-server-tacacs)# key key_2
Device(config-server-tacacs)# end
```

The **broadcast** keyword causes “start” and “stop” accounting records for network connection calls having DNIS number 7777 to be sent simultaneously to server 10.0.0.1 in the group isp and to server 172.0.0.1 in the group isp_customer. If server 10.0.0.1 is unavailable, failover to server 10.0.0.2 occurs. If server 172.0.0.1 is unavailable, no failover occurs because backup servers are not configured for the group isp_customer.

## Example: AAA Session MIB

The following example shows how to set up the AAA session MIB feature to disconnect authenticated client connections for PPP users:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication ppp default group radius
Device(config)# aaa authorization network default group radius
Device(config)# aaa accounting network default start-stop group radius
Device(config)# aaa session-mib disconnect
Device(config)# end
```

# Additional References for Configuring Accounting

The following sections provide references related to the Configuring Accounting feature.

### MIBs

| MIB | MIBs Link |
|---|---|
| • CISCO-AAA-SESSION-MIB | To locate and download MIBs for selected platforms, Cisco IOS XE software releases , and feature sets, use Cisco MIB Locator found at the following URL:<br>http://www.cisco.com/go/mibs |

### RFCs

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/techsupport |

# Feature History for Configuring Accounting

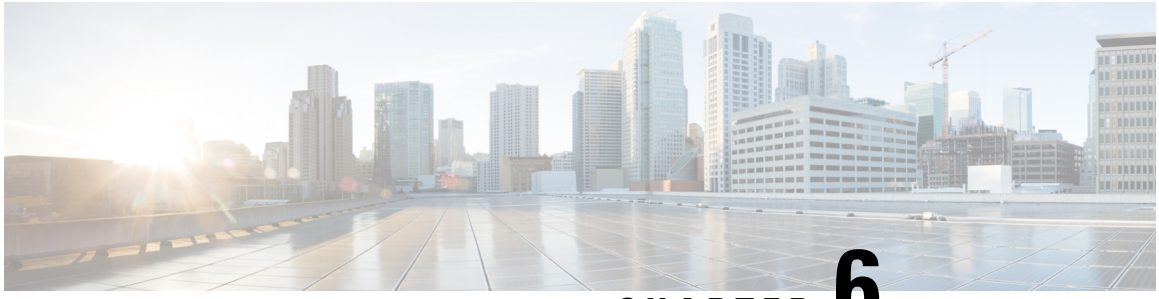
This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | AAA Broadcast Accounting | AAA broadcast accounting allows accounting information to be sent to multiple AAA servers at the same time; that is, accounting information can be broadcast to one or more AAA servers simultaneously.<br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | AAA Broadcast Accounting | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Everest 16.5.1a | AAA Session MIB | The AAA session MIB feature allows customers to monitor and terminate their authenticated client connections using SNMP.<br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | AAA Session MIB | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Connection Accounting | Connection accounting provides information about all outbound connections made from the network access server, such as Telnet, local-area transport, TN3270, packet assembler/disassembler (PAD), and rlogin.<br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Connection Accounting | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Everest 16.5.1a | AAA Interim Accounting | AAA interim accounting allows accounting records to be sent to the accounting server every time there is new accounting information to report, or on a periodic basis.<br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | AAA Interim Accounting | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.



CHAPTER 6

# Configuring Local Authentication and Authorization


- How to Configure Local Authentication and Authorization, on page 117
- Monitoring Local Authentication and Authorization, on page 119
- Feature History for Local Authentication and Authorization, on page 119


# How to Configure Local Authentication and Authorization

## Configuring the Switch for Local Authentication and Authorization

You can configure AAA to operate without a server by setting the switch to implement AAA in local mode. The switch then handles authentication and authorization. No accounting is available in this configuration.

**Note** To secure the switch for HTTP access by using AAA methods, you must configure the switch with the **ip http authentication aaa** global configuration command. Configuring AAA authentication does not secure the switch for HTTP access by using AAA methods.

Follow these steps to configure AAA to operate without a server by setting the switch to implement AAA in local mode:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authentication login default local**<br>**Example:**<br>`Device(config)# aaa authentication login default local` | Sets the login authentication to use the local username database. The **default** keyword applies the local user database authentication to all ports. |
| **Step 5** | **aaa authorization exec default local**<br>**Example:**<br>`Device(config)# aaa authorization exec default local` | Configures user AAA authorization, check the local database, and allow the user to run an EXEC shell. |
| **Step 6** | **aaa authorization network default local**<br>**Example:**<br>`Device(config)# aaa authorization network default local` | Configures user AAA authorization for all network-related service requests. |
| **Step 7** | **username** *name* [**privilege** *level*] {**password** *encryption-type password*}<br>**Example:**<br>`Device(config)# username your_user_name privilege 1 password 7 secret567` | Enters the local database, and establishes a username-based authentication system.<br>Repeat this command for each user.<br>• For *name*, specify the user ID as one word. Spaces and quotation marks are not allowed.<br>• (Optional) For *level*, specify the privilege level the user has after gaining access. The range is 0 to 15. Level 15 gives privileged EXEC mode access. Level 0 gives user EXEC mode access.<br>• For *encryption-type*, enter 0 to specify that an unencrypted password follows. Enter 7 to specify that a hidden password follows.<br>• For *password*, specify the password the user must enter to gain access to the switch. The password must be from 1 to |

| | Command or Action | Purpose |
|---|---|---|
| | | 25 characters, can contain embedded spaces, and must be the last option specified in the **username** command. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Monitoring Local Authentication and Authorization

To display Local Authentication and Authorization configuration, use the **show running-config** command in privileged EXEC mode.

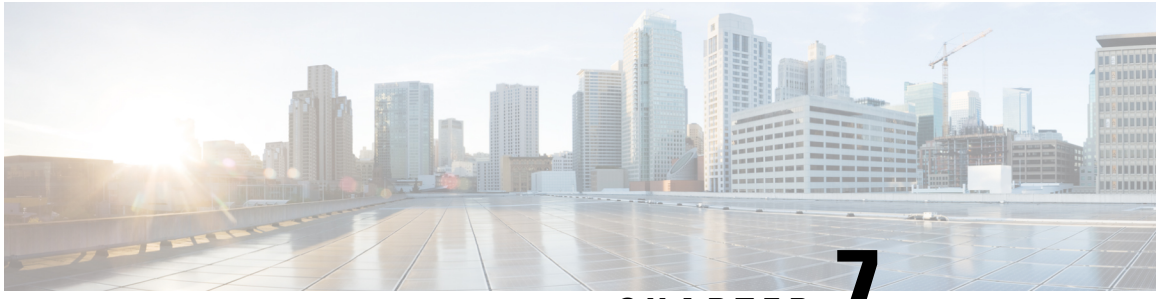
# Feature History for Local Authentication and Authorization

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Local Authentication and Authorization | This feature helps AAA to operate without a server by setting the device to implement AAA in local mode.<br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.



CHAPTER 7

# Configuring AAA Dead-Server Detection

The AAA Dead-Server Detection feature allows you to configure the criteria to be used to mark a RADIUS server as dead. If no criteria are explicitly configured, the criteria are computed dynamically on the basis of the number of outstanding transactions. Using this feature will result in less deadtime and quicker packet processing.


- Prerequisites for AAA Dead-Server Detection, on page 121
- Restrictions for AAA Dead-Server Detection, on page 121
- Information About AAA Dead-Server Detection, on page 121
- How to Configure AAA Dead-Server Detection, on page 122
- Configuration Examples for AAA Dead-Server Detection, on page 124
- Feature History for AAA Dead-Server Detection, on page 125


## Prerequisites for AAA Dead-Server Detection

- You must have access to a RADIUS server.
- You should be familiar with configuring a RADIUS server.
- You should be familiar with configuring authentication, authorization, and accounting (AAA).
- Before a server can be marked as dead, you must first configure the **radius-server deadtime** command. If this command is not configured, even if the criteria are met for the server to be marked as dead, the server state will be in the up state.

## Restrictions for AAA Dead-Server Detection

- Original transmissions are not counted in the number of consecutive timeouts that must occur on the device before the server is marked as dead; only the number of retransmissions are counted.

## Information About AAA Dead-Server Detection

This section provides information about the AAA Dead-Server Detection feature.

# Criteria for Marking a RADIUS Server As Dead

The AAA Dead-Server Detection feature allows you to determine the criteria that are used to mark a RADIUS server as dead. That is, you can configure the minimum amount of time, in seconds, that must elapse from the time that the device last received a valid packet from the RADIUS server to the time the server is marked as dead. If a packet has not been received since the device booted, and there is a timeout, the time criterion will be treated as though it has been met.

In addition, you can configure the number of consecutive timeouts that must occur on the device before the RADIUS server is marked as dead. If the server performs both authentication and accounting, both types of packets are included in the number. Improperly constructed packets are counted as though they are timeouts. Only retransmissions are counted, not the initial transmission. (Each timeout causes one retransmission to be sent.)

**Note** Both the time criterion and the tries criterion must be met for the server to be marked as dead.

The RADIUS dead-server detection configuration will result in the prompt detection of RADIUS servers that have stopped responding. This configuration will also result in the avoidance of servers being improperly marked as dead when they are swamped (responding slowly) and the avoidance of the state of servers being rapidly changed from dead to live to dead again. This prompt detection of nonresponding RADIUS servers and the avoidance of swamped and dead-to-live-to-dead-again servers will result in less deadtime and quicker packet processing.

Each AAA RADIUS global and server groups can have its own deadtime configured. The deadtime configured on the server group takes precedence over the global deadtime configuration. When a deadtime is configured on any AAA RADIUS server group, it clears the existing dead timer on all global server groups that are marked as dead, and not just the specified server group.

# How to Configure AAA Dead-Server Detection

This section describes how to configure AAA dead-server detection.

## Configuring AAA Dead-Server Detection

To configure AAA Dead-Server Detection, perform the following steps.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables the AAA access control model. |
| **Step 4** | **radius-server deadtime** *minutes*<br>**Example:**<br>`Device(config)# radius-server deadtime 5` | Improves RADIUS response times when some servers might be unavailable and causes the unavailable servers to be skipped immediately. |
| **Step 5** | **radius-server dead-criteria** [**time** *seconds*] [**tries** *number-of-tries*]<br>**Example:**<br>`Device(config)# radius-server dead-criteria time 5 tries 4` | Forces one or both of the criteria, used to mark a RADIUS server as dead, to be the indicated constant. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config)# end` | Returns to privileged EXEC mode. |
| **Step 7** | **show running-config**<br>**Example:**<br>`Device# show running-config` | Verifies your configuration.<br>After you have configured AAA Dead-Server Detection, you should verify your configuration using this command. This verification is especially important if you have used the **no** form of the **radius-server dead-criteria** command. The output of this command must show the same values in the **Dead Criteria Details** field that you configured using the **radius-server dead-criteria** command. |

# Verifying AAA Dead-Server Detection

To verify your AAA Dead-Server Detection configuration, perform the following steps. The **show** and **debug** commands may be used in any order.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:** | Enables privileged EXEC mode.<br>Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device> enable` | |
| **Step 2** | **debug aaa dead-criteria transactions**<br>**Example:**<br>`Device# debug aaa dead-criteria transactions` | Displays AAA dead-criteria transaction values. |
| **Step 3** | **show aaa dead-criteria**<br>**Example:**<br>`Device# show aaa dead-criteria` | Displays dead-criteria information for a AAA server. |
| **Step 4** | **show aaa servers** [**private** \| **public**]<br>**Example:**<br>`Device# show aaa server private` | Displays the status and number of packets that are sent to and received from all public and private authentication, authorization, and accounting (AAA) RADIUS servers.<br>• The **private** keyword optionally displays the AAA servers only.<br>• The **public** keyword optionally displays the AAA servers only. |

# Configuration Examples for AAA Dead-Server Detection

The following sections show configuration examples of AAA dead-server detection:

## Example: Configuring AAA Dead-Server Detection

The following example shows that the device will be considered dead after 5 seconds and four tries:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# radius-server deadtime 5
Device(config)# radius-server dead-criteria time 5 tries 4
```

The following output example shows dead-criteria transaction information for a particular server group:

```
Device> enable
Device# debug aaa dead-criteria transactions

AAA Transaction debugs debugging is on
*Nov 14 23:44:17.403: AAA/SG/TRANSAC: Computed Retransmit Tries: 22, Current Max Tries: 22
*Nov 14 23:44:17.403: AAA/SG/TRANSAC: Computed Dead Detect Interval: 25s, Current Max
Interval: 25s
*Nov 14 23:44:17.403: AAA/SG/TRANSAC: Estimated Outstanding Transactions: 6, Current Max
Transactions: 6
```

The following output example shows that dead-server-detection information has been requested for a RADIUS server at the IP address 172.19.192.80:

```
Device> enable
Device# show aaa dead-criteria radius 172.19.192.80 radius

RADIUS Server Dead Criteria:
============================
Server Details:
    Address : 172.19.192.80
    Auth Port : 1645
    Acct Port : 1646
Server Group : radius
Dead Criteria Details:
    Configured Retransmits : 62
    Configured Timeout : 27
    Estimated Outstanding Transactions: 5
    Dead Detect Time : 25s
    Computed Retransmit Tries: 22
    Statistics Gathered Since Last Successful Transaction
=====================================================
Max Computed Outstanding Transactions: 5
Max Computed Dead Detect Time: 25s
Max Computed Retransmits : 22
```

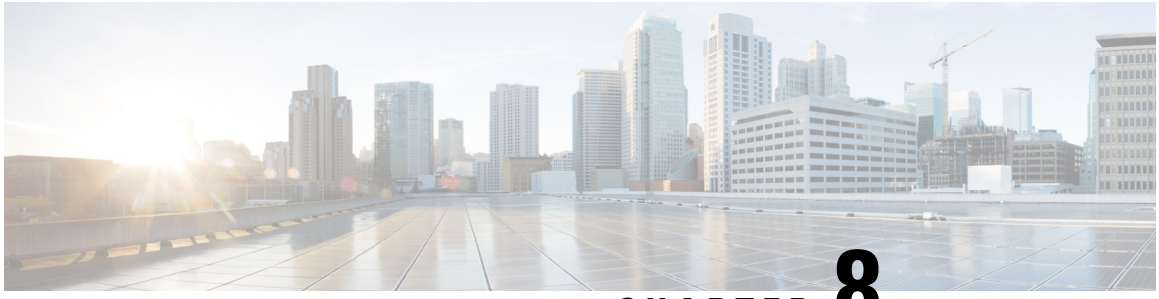
# Feature History for AAA Dead-Server Detection

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | AAA Dead-Server Detection | This feature allows you to configure the criteria to be used to mark a RADIUS server as dead.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalsyt 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | AAA Dead-Server Detection | Support for this feature was introduced on the C9500-24Y4C, C9500-32C, C9500-32QC, and C9500-48Y4C models of the Cisco Catalsyt 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.



CHAPTER 8

# Configuring TACACS+


- Prerequisites for TACACS+, on page 127
- Information About TACACS+, on page 128
- How to Configure TACACS+, on page 131
- Monitoring TACACS+, on page 139
- Additional References for TACACS+, on page 139
- Feature History for TACACS+, on page 140


## Prerequisites for TACACS+

The following are the prerequisites for set up and configuration of switch access with TACACS+ (must be performed in the order presented):

1. Configure the switches with the TACACS+ server addresses.
2. Set an authentication key.
3. Configure the key from Step 2 on the TACACS+ servers.
4. Enable authentication, authorization, and accounting (AAA).
5. Create a login authentication method list.
6. Apply the list to the terminal lines.
7. Create an authorization and accounting method list.

The following are the prerequisites for controlling switch access with TACACS+:

- You must have access to a configured TACACS+ server to configure TACACS+ features on your switch. Also, you must have access to TACACS+ services maintained in a database on a TACACS+ daemon typically running on a LINUX or Windows workstation.
- You need a system running the TACACS+ daemon software to use TACACS+ on your switch.
- To use TACACS+, it must be enabled.
- Authorization must be enabled on the switch to be used.
- Users must first successfully complete TACACS+ authentication before proceeding to TACACS+ authorization.

- To use any of the AAA commands listed in this section or elsewhere, you must first enable AAA with the **aaa new-model** command.
- At a minimum, you must identify the host or hosts maintaining the TACACS+ daemon and define the method lists for TACACS+ authentication. You can optionally define method lists for TACACS+ authorization and accounting.
- The method list defines the types of authentication to be performed and the sequence in which they are performed; it must be applied to a specific port before any of the defined authentication methods are performed. The only exception is the default method list (which, by coincidence, is named *default*). The default method list is automatically applied to all ports except those that have a named method list explicitly defined. A defined method list overrides the default method list.
- Use TACACS+ for privileged EXEC access authorization if authentication was performed by using TACACS+.
- Use the local database if authentication was not performed by using TACACS+.

# Information About TACACS+

## TACACS+ and Switch Access

This section describes TACACS+. TACACS+ provides detailed accounting information and flexible administrative control over the authentication and authorization processes. It is facilitated through authentication, authorization, accounting (AAA) and can be enabled only through AAA commands.

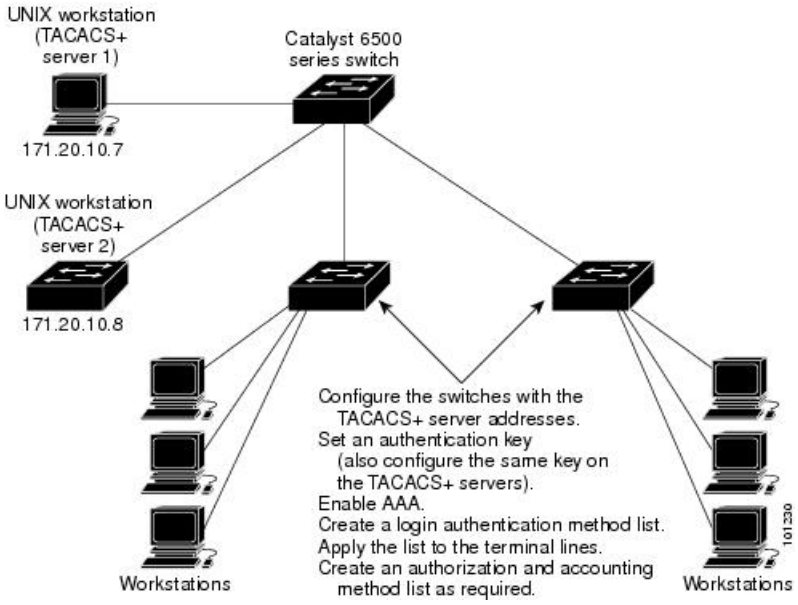
## TACACS+ Overview

TACACS+ is a security application that provides centralized validation of users attempting to gain access to your switch.

TACACS+ provides for separate and modular authentication, authorization, and accounting facilities. TACACS+ allows for a single access control server (the TACACS+ daemon) to provide each service—authentication, authorization, and accounting—independently. Each service can be tied into its own database to take advantage of other services available on that server or on the network, depending on the capabilities of the daemon.

The goal of TACACS+ is to provide a method for managing multiple network access points from a single management service. Your switch can be a network access server along with other Cisco routers and access servers.

*Figure 9: Typical TACACS+ Network Configuration*




TACACS+, administered through the AAA security services, can provide these services:

- Authentication—Provides complete control of authentication through login and password dialog, challenge and response, and messaging support.

  The authentication facility can conduct a dialog with the user (for example, after a username and password are provided, to challenge a user with several questions, such as home address, mother's maiden name, service type, and social security number). The TACACS+ authentication service can also send messages to user screens. For example, a message could notify users that their passwords must be changed because of the company's password aging policy.

- Authorization—Provides fine-grained control over user capabilities for the duration of the user's session, including but not limited to setting autocommands, access control, session duration, or protocol support. You can also enforce restrictions on what commands a user can execute with the TACACS+ authorization feature.

- Accounting—Collects and sends information used for billing, auditing, and reporting to the TACACS+ daemon. Network managers can use the accounting facility to track user activity for a security audit or to provide information for user billing. Accounting records include user identities, start and stop times, executed commands (such as PPP), number of packets, and number of bytes.

The TACACS+ protocol provides authentication between the switch and the TACACS+ daemon, and it ensures confidentiality because all protocol exchanges between the switch and the TACACS+ daemon are encrypted.

# TACACS+ Operation

When a user attempts a simple ASCII login by authenticating to a switch using TACACS+, this process occurs:

1. When the connection is established, the switch contacts the TACACS+ daemon to obtain a username prompt to show to the user. The user enters a username, and the switch then contacts the TACACS+

daemon to obtain a password prompt. The switch displays the password prompt to the user, the user enters a password, and the password is then sent to the TACACS+ daemon.

TACACS+ allows a dialog between the daemon and the user until the daemon receives enough information to authenticate the user. The daemon prompts for a username and password combination, but can include other items, such as the user's mother's maiden name.

2. The switch eventually receives one of these responses from the TACACS+ daemon:
   - ACCEPT—The user is authenticated and service can begin. If the switch is configured to require authorization, authorization begins at this time.
   - REJECT—The user is not authenticated. The user can be denied access or is prompted to retry the login sequence, depending on the TACACS+ daemon.
   - ERROR—An error occurred at some time during authentication with the daemon or in the network connection between the daemon and the switch. If an ERROR response is received, the switch typically tries to use an alternative method for authenticating the user.
   - CONTINUE—The user is prompted for additional authentication information.

   After authentication, the user undergoes an additional authorization phase if authorization has been enabled on the switch. Users must first successfully complete TACACS+ authentication before proceeding to TACACS+ authorization.
3. If TACACS+ authorization is required, the TACACS+ daemon is again contacted, and it returns an ACCEPT or REJECT authorization response. If an ACCEPT response is returned, the response contains data in the form of attributes that direct the EXEC or NETWORK session for that user and the services that the user can access:
   - Telnet, Secure Shell (SSH), rlogin, or privileged EXEC services
   - Connection parameters, including the host or client IP address, access list, and user timeouts

# Method List

A method list defines the sequence and methods to be used to authenticate, to authorize, or to keep accounts on a user. You can use method lists to designate one or more security protocols to be used, thus ensuring a backup system if the initial method fails. The software uses the first method listed to authenticate, to authorize, or to keep accounts on users; if that method does not respond, the software selects the next method in the list. This process continues until there is successful communication with a listed method or the method list is exhausted.

# TACACS+ Configuration Options

You can configure the switch to use a single server or AAA server groups to group existing server hosts for authentication. You can group servers to select a subset of the configured server hosts and use them for a particular service. The server group is used with a global server-host list and contains the list of IP addresses of the selected server hosts.

## TACACS+ Login Authentication

A method list describes the sequence and authentication methods to be queried to authenticate a user. You can designate one or more security protocols to be used for authentication, thus ensuring a backup system for authentication in case the initial method fails. The software uses the first method listed to authenticate users; if that method fails to respond, the software selects the next authentication method in the method list. This process continues until there is successful communication with a listed authentication method or until all defined methods are exhausted. If authentication fails at any point in this cycle—meaning that the security server or local username database responds by denying the user access—the authentication process stops, and no other authentication methods are attempted.

## TACACS+ Authorization for Privileged EXEC Access and Network Services

AAA authorization limits the services available to a user. When AAA authorization is enabled, the switch uses information retrieved from the user's profile, which is located either in the local user database or on the security server, to configure the user's session. The user is granted access to a requested service only if the information in the user profile allows it.

## TACACS+ Accounting

The AAA accounting feature tracks the services that users are accessing and the amount of network resources that they are consuming. When AAA accounting is enabled, the switch reports user activity to the TACACS+ security server in the form of accounting records. Each accounting record contains accounting attribute-value (AV) pairs and is stored on the security server. This data can then be analyzed for network management, client billing, or auditing.

## Default TACACS+ Configuration

TACACS+ and AAA are disabled by default.

To prevent a lapse in security, you cannot configure TACACS+ through a network management application. When enabled, TACACS+ can authenticate users accessing the switch through the CLI.

**Note** Although TACACS+ configuration is performed through the CLI, the TACACS+ server authenticates HTTP connections that have been configured with a privilege level of 15.

# How to Configure TACACS+

This section describes how to configure your switch to support TACACS+.

## Identifying the TACACS+ Server Host and Setting the Authentication Key

Follow these steps to identify the TACACS+ server host and set the authentication key:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **tacacs server** *server-name*<br>**Example:**<br>`Device(config)# tacacs server yourserver` | Identifies the IP host or hosts maintaining a TACACS+ server. Enter this command multiple times to create a list of preferred hosts. The software searches for hosts in the order in which you specify them.<br>For *server-name*, specify the server name. |
| **Step 4** | **address** {**ipv4** \| **ipv6**} *ip address*<br>**Example:**<br>`Device(config-server-tacacs)# address ipv4 10.0.1.12` | Configures the IP address for the TACACS server. |
| **Step 5** | **key** [*encryption-type*] [*key-string*]<br>**Example:**<br>`Device(config-server-tacacs)# key 0 auth-key` | Sets the authentication encryption key used for all TACACS+ communications between the access server and the TACACS+ daemon. This encryption key must match the key used on the TACACS+ daemon.<br>*encryption-type* is optional, and if nothing is specified it is considered as clear text. Enter **0** to specify that an unencrypted key will follow. Enter **6** to specify that an encrypted key will follow. Enter **7** to specify that a hidden key will follow. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(config-server-tacacs)# exit` | Exits the TACACS server mode and enters the global configuration mode. |
| **Step 7** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **aaa group server tacacs+** *group-name*<br>**Example:**<br>`Device(config)# aaa group server tacacs+ your_server_group` | (Optional) Defines the AAA server-group with a group name, and enters server group configuration mode. |
| **Step 9** | **server name** *server-name*<br>**Example:**<br>`Device(config-sg-tacacs)# server name yourserver` | (Optional) Associates a particular TACACS+ server with the defined server group. Repeat this step for each TACACS+ server in the AAA server group.<br>Each server in the group must be previously defined in Step 3. |
| **Step 10** | **end**<br>**Example:**<br>`Device(config-sg-tacacs)# end` | Exits server group configuration mode and returns to privileged EXEC mode. |

# Configuring TACACS+ Login Authentication

Follow these steps to configure TACACS+ login authentication:

### Before you begin

To configure AAA authentication, you define a named list of authentication methods and then apply that list to various ports.

**Note** To secure the device for HTTP access by using AAA methods, you must configure the device with the **ip http authentication aaa** global configuration command. Configuring AAA authentication does not secure the device for HTTP access by using AAA methods.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authentication login** {**default** \| *list-name*} *method1* [*method2...*]<br>**Example:**<br>`Device(config)# aaa authentication login default tacacs+ local` | Creates a login authentication method list.<br>• To create a default list that is used when a named list is *not* specified in the **login authentication** command, use the **default** keyword followed by the methods that are to be used in default situations. The default method list is automatically applied to all ports.<br>• For *list-name*, specify a character string to name the list you are creating.<br>• For *method1...*, specify the actual method the authentication algorithm tries. The additional methods of authentication are used only if the previous method returns an error, not if it fails.<br><br>Select one of these methods:<br>• *enable*—Use the enable password for authentication. Before you can use this authentication method, you must define an enable password by using the **enable** *password* global configuration command.<br>• *group tacacs+*—Uses TACACS+ authentication. Before you can use this authentication method, you must configure the TACACS+ server.<br>• *line* —Use the line password for authentication. Before you can use this authentication method, you must define a line password. Use the **password** *password* line configuration command.<br>• *local*—Use the local username database for authentication. You must enter username information in the database. Use the **username** *password* global configuration command.<br>• *local-case*—Use a case-sensitive local username database for authentication. You must enter username information in the |

| | Command or Action | Purpose |
|---|---|---|
| | | database by using the **username** *name* **password** global configuration command.<br>• *none*—Do not use any authentication for login. |
| **Step 5** | **line** [**console** \| **tty** \| **vty**] *line-number* [*ending-line-number*]<br>**Example:**<br>`Device(config)# line 2 4` | Enters line configuration mode, and configures the lines to which you want to apply the authentication list. |
| **Step 6** | **login authentication** {**default** \| *list-name*}<br>**Example:**<br>`Device(config-line)# login authentication default` | Applies the authentication list to a line or set of lines.<br>• If you specify **default**, use the default list created with the **aaa authentication login** command.<br>• For *list-name*, specify the list created with the **aaa authentication login** command. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-line)# end` | Exits line configuration mode and returns to privileged EXEC mode. |

# Configuring TACACS+ Authorization for Privileged EXEC Access and Network Services

You can use the **aaa authorization** global configuration command with the **tacacs+** keyword to set parameters that restrict a user's network access to privileged EXEC mode.

**Note** Authorization is bypassed for authenticated users who log in through the CLI even if authorization has been configured.

Follow these steps to specify TACACS+ authorization for privileged EXEC access and network services:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:** | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device> enable` | |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa authorization network***authorization-list* **tacacs+**<br>**Example:**<br>`Device(config)# aaa authorization network list1 tacacs+` | Configures the switch for user TACACS+ authorization for all network-related service requests. |
| **Step 4** | **aaa authorization exec** *default***tacacs+**<br>**Example:**<br>`Device(config)# aaa authorization exec default tacacs+` | Configures the switch for user TACACS+ authorization if the user has privileged EXEC access.<br>The **exec** keyword might return user profile information (such as **autocommand** information). |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Starting TACACS+ Accounting

Follow these steps to start TACACS+ Accounting:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **aaa accounting network** *authorization-list* **start-stop tacacs+**<br>**Example:**<br>`Device(config)# aaa accounting network list1 start-stop tacacs+` | Enables TACACS+ accounting for all network-related service requests. |
| **Step 4** | **aaa accounting exec** *default* **start-stop tacacs+**<br>**Example:**<br>`Device(config)# aaa accounting exec default start-stop tacacs+` | Enables TACACS+ accounting to send a start-record accounting notice at the beginning of a privileged EXEC process and a stop-record at the end. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

### What to do next

To establish a session with a device if the AAA server is unreachable, use the **aaa accounting system guarantee-first** command. It guarantees system accounting as the first record, which is the default condition. In some situations, users might be prevented from starting a session on the console or terminal connection until after the system reloads, which can take more than 3 minutes.

To establish a console or Telnet session with the router if the AAA server is unreachable when the router reloads, use the **no aaa accounting system guarantee-first** command.

# Establishing a Session with a Device if the AAA Server is Unreachable

To establishing a session with a device if the AAA server is unreachable, use the **aaa accounting system guarantee-first** command. It guarantees system accounting as the first record, which is the default condition. In some situations, users might be prevented from starting a session on the console or terminal connection until after the system reloads, which can take more than 3 minutes.

To establish a console or Telnet session with the device if the AAA server is unreachable when the device reloads, use the **no aaa accounting system guarantee-first** command.

# Configuring TACACS Source-Interface Under a TACACS Server-Group

The TACACS source-interface can be configured under a TACACS server-group in either of the following methods:

- Configure a TACACS source-interface under the TACACS server-group using the **ip tacacs source-interface** *interface-name* command.

- Configure a VRF using the **vrf** *vrf-name* command under the TACACS server-group, and then associate the configured VRF globally to a source-interface using the **ip tacacs source interface** *interface-name* **vrf** *vrf-name* command.

Priority will be given to the source-interface under the server-group configuration in case both methods are configured.

To configure TACACS source-interface under a TACACS server-group, perform the following:

### Before you begin

You must configure a VRF routing table and associate VRF to an interface

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | { **ip** \| **ipv6** }**tacacs source-interface** *interface-number* **vrf** *vrf-name*<br>**Example:**<br>`Device(config)# ip tacacs source-interface GigabitEthernet1/0/23 vrf vrf17` | Forces TACACS to use the IP address of a specified interface for all outgoing TACACS packets, and enables the specification on a per-VRF basis.<br>• *interface-name*: Specifies the name of the interface that TACACS+ uses for all of its outgoing packets.<br>• **vrf** *vrf-name*: Specifies the per-VRF configuration. |
| **Step 4** | **aaa group server tacacs** *group_name*<br>**Example:**<br>`Device(config-sg-tacacs+)# aa group server tacacs rad-grp` | Groups different TACACS server hosts into distinct lists and distinct methods and enters server-group configuration mode. |
| **Step 5** | **ip vrf forwarding** *vrf-name*<br>**Example:**<br>`Device(config-sg-tacacs+)# ip vrf forwarding vrf17` | (Optional) Configures a VRF for the interface. |
| **Step 6** | { **ip** \| **ipv6** } **tacacs source-interface** *interface-number*<br>**Example:** | (Optional) Forces TACACS+ to use the IP address of a specified interface for all outgoing TACACS packets from the TACACS+ group server. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-sg-tacacs+)# ip tacacs source-interface loopback0` | *interface-name*: Specifies the name of the interface that TACACS uses for all of its outgoing packets. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-sg-tacacs+)# end` | Returns to privileged EXEC mode. |

# Monitoring TACACS+

***Table 8: Commands for Displaying TACACS+ Information***

| Command | Purpose |
|---|---|
| **show tacacs** | Displays TACACS+ server statistics. |

# Additional References for TACACS+

### Related Documents

| Related Topic | Document Title |
|---|---|
| AAA configuration | Configuring Authentication, Configuring Authorization, and Configuring Accounting chapters of the *Security Configuration Guide* |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/support |

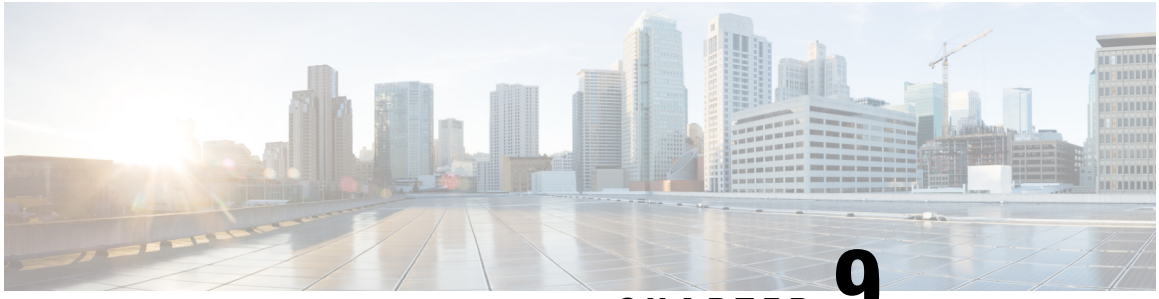
# Feature History for TACACS+

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | TACACS+ | TACACS+ provides detailed accounting information and flexible administrative control over authentication and authorization processes. TACACS+ is facilitated through AAA and can be enabled only through AAA commands.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | TACACS+ | Support for this feature was introduced on the C9500-24Y4C, C9500-32C, C9500-32QC, and C9500-48Y4C models of the Cisco Catalsyt 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.



CHAPTER 9

# Device Sensor

The Device Sensor feature is used to gather raw endpoint data from network devices using protocols such as Cisco Discovery Protocol (CDP), Link Layer Discovery Protocol (LLDP), Dynamic Host Configuration Protocol (DHCP), DHCP version 6, and multicast DNS (mDNS). The endpoint data that is gathered is made available to registered clients in the context of an access session.


- Restrictions for Device Sensor, on page 141
- Information About Device Sensor, on page 141
- How to Configure Device Sensor, on page 143
- Configuration Examples for Device Sensor , on page 149
- Feature History for Device Sensor, on page 150


# Restrictions for Device Sensor

- Only Cisco Discovery Protocol (CDP), Link Layer Discovery Protocol (LLDP), Dynamic Host Configuration Protocol (DHCP), Dynamic Host Configuration Protocol ersion 6 (DHCPv6), and multicast DNS (mDNS) protocols are supported.
- The session limit for profiling ports is 32.
- The length of one Type-Length-Value (TLV) must not be more than 1024 and the total length of TLVs (combined length of TLVs) of all protocols must not be more than 4096.
- The sensor profiles devices that are only one hop away.
- The Device Sensor feature is enabled by default, but cannot be disabled. Disabling device classifier using **no device classifier** command in global configuration mode does not disable device sensor. This is because device sensor is independent of IP device tracking and device classifier.

# Information About Device Sensor

## Device Sensor

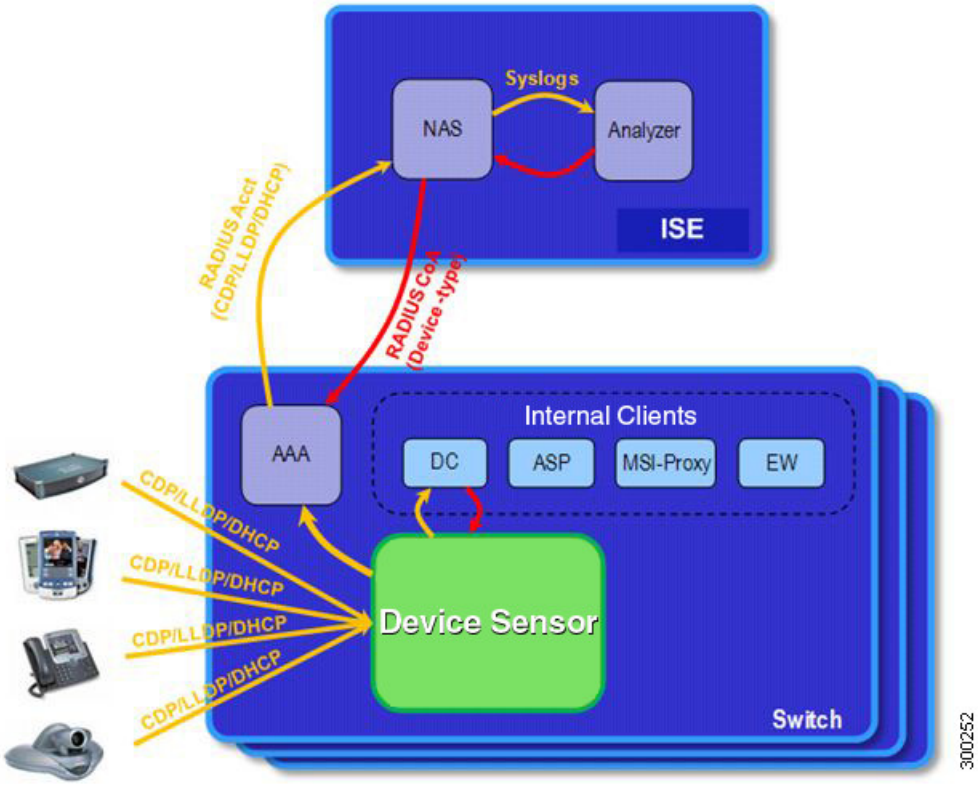
The device sensor is used to gather raw endpoint data from network devices. The endpoint information that is gathered helps in completing the profiling capability of devices. Profiling is the determination of the endpoint

type based on information gleaned from various protocol packets from an endpoint during its connection to a network.

The profiling capability consists of two parts:

- Collector—Gathers endpoint data from network devices.
- Analyzer—Processes the data and determines the type of device.

The device sensor represents the embedded collector functionality. The illustration below shows the Cisco sensor in the context of the profiling system and also features other possible clients of the sensor.




A device with sensor capability gathers endpoint information from network devices using protocols such as Cisco Discovery Protocol, LLDP, DHCPv6, mDNS and DHCP, subject to statically configured filters, and makes this information available to its registered clients in the context of an access session. An access session represents an endpoint's connection to the network device.

The device sensor has internal and external clients. The internal clients include components such as the embedded Device Classifier (local analyzer), ATM switch processor (ASP), MSI-Proxy, and EnergyWise (EW). The external client, that is the Identity Services Engine (ISE) analyzer, will use RADIUS accounting to receive additional endpoint data.

Client notifications and accounting messages containing profiling data along with the session events and other session-related data, such as the MAC address and the ingress port, are generated and sent to the internal and external clients (ISE). By default, for each supported peer protocol, client notifications and accounting events are only generated where an incoming packet includes a TLV that has not previously been received in the context of a given session. You can enable client notifications and accounting events for all TLV changes,

where either a new TLV has been received or a previously received TLV has been received with a different value using CLI commands.

The device sensor's port security protects the switch from consuming memory and crashing during deliberate or unintentional denial-of-service (DoS) type attacks. The sensor limits the maximum device monitoring sessions to 32 per port (access ports and trunk ports). In case of lack of activity from hosts, the idle session time is 12 hours.

# How to Configure Device Sensor

The device sensor is enabled by default.

The following tasks are applicable only if you want to configure the sensor based on your specific requirements.

**Note** If you do not perform these configuration tasks, then the following TLVs are included by default:

- Cisco Discovery Protocol filter—secondport-status-type and powernet-event-type (type 28 and 29).
- LLDP filter—organizationally-specific (type 127).
- DHCP filter—message-type (type 53).

## Enabling Accounting Augmentation

Perform this task to add device sensor protocol data to accounting records.

### Before you begin

- The device must be in IBNS 1.0 mode before performing this task.
- For the sensor protocol data to be added to the accounting messages, you must enable session accounting by using the standard authentication, authorization, and accounting (AAA), and RADIUS configuration commands.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **device-sensor accounting**<br>**Example:**<br>`Device(config)# device-sensor accounting` | Enables the addition of sensor protocol data to accounting records and also enables the generation of additional accounting events when new sensor data is detected. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring Protocol Attributes in Access and Accounting Requests

Perform this task to create an attribute filter-list and to bind an attribute filter-list with authentication and accounting requests.

### Before you begin

The device must be in IBNS 2.0 mode before performing this task.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device(config)# configure terminal` | Enters global configuration mode. |
| **Step 3** | **access-session attributes filter-list list** *list-name*<br>**Example:**<br>`Device(config)# access-session attributes filter-list list mylist` | Adds access-session protocol data to accounting and authentication records and enters common filter list configuration mode. The **filter-list** keyword configures a sensor protocol filter list to accounting and authentication records. |
| **Step 4** | {**cdp** \| **dhcp** \| **dhcpv6** \| **http** \| **lldp** \| **vlan-id**}<br>**Example:**<br>`Device(config-com-filter-list)# dhcp` | Includes the specified protocol for the attribute. |
| **Step 5** | **exit**<br>**Example:**<br>`Device(config-com-filter-list)# exit` | Exits common filter list configuration mode and returns to global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **access-session** {**accounting** \| **authentication**} **attributes filter-spec include list** *list-name*<br>**Example:**<br>`Device(config)# access-session authentication attributes filter-spec include list mylist` | Configures a sensor protocol filter specification, and binds an attribute filter list with accounting and authentication records. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Creating a Protocol Filter

Perform this task to create a CDP, LLDP, DHCP, mDNS, or DHCPv6 filter containing TLVs that can be included or excluded in the device sensor output.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode. Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **device-sensor**<br>{**filter-list***cdp* \| *dhcp* \| *dhcpv6* \| *mdns* \| *lldp*}<br>{**list***tlv-list-name*}<br>**Example:**<br>`Device(config)# device-sensor filter-list cdp list cdp-list` | Applies a sensor protocol filter list and enters configuration mode, where you can configure individual TLVs.<br>• **cdp** - Applies a Cisco Discovery Protocol TLV filter list.<br>• **lldp** - Applies an Link Layer Discovery protocol TLV filter list.<br>• **dhcp** - Applies a Dynamic Host Configuration Protocol TLV filter list.<br>• **dhcpv6** - Applies a Dynamic Host Configuration Protocol version 6 TLV filter list.<br>• **mdns** - Applies a Multicast DNS Protocol TLV filter list. |

| | Command or Action | Purpose |
|---|---|---|
| | | • **list***list-name* - Specifies the protocol TLV filter list name. |
| **Step 4** | **option** {**name** *option-name* \| **number** *option-number*}<br>**Example:**<br>`Device(config-sensor-cdplist)# tlv number 10` | This step applies only to DHCP protocol. Adds individual DHCP options to the option list.<br>You can delete the option list without individually removing options from the list by using the **no device-sensor filter-list dhcp list** *option-list-name* command.<br>• You can delete the TLV list without individually removing TLVs from the list by using the **no device-sensor filter-list cdp list** *tlv-list-name* command. |
| **Step 5** | **tlv** {**name** *tlv-name* \| **number** *tlv-number*}<br>**Example:**<br>`Device(config-sensor-cdplist)# tlv number 10` | Adds individual Cisco Discovery Protocol TLVs to the TLV list.<br>• You can delete the TLV list without individually removing TLVs from the list by using the **no device-sensor filter-list cdp list** *tlv-list-name* command. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-sensor-cdplist)# end` | Returns to privileged EXEC mode. |

# Applying a Protocol Filter to the Sensor Output

Perform this task to apply a Cisco Discovery Protocol, LLDP, or DHCP filter to the sensor output. Session notifications are sent to internal sensor clients and accounting requests.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode. Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **device-sensor filter-spec** {**cdp** \| **dhcp** \| **lldp**} {**exclude** {**all** \| **list** *list-name*} \| **include list** *list-name*}<br>**Example:**<br>`Device(config)# device-sensor filter-spec cdp include list list1` | Applies a specific protocol filter containing a list of TLV fields to the device sensor output.<br>• **cdp**—Applies a Cisco Discovery Protocol TLV filter list to the device sensor output.<br>• **lldp**—Applies an LLDP TLV filter list to the device sensor output.<br>• **dhcp**—Applies a DHCP TLV filter list to the device sensor output.<br>• **dhcpv6**—Applies a DHCPv6 TLV filter list to the device sensor output.<br>• **mdns**—Applies an mDNS TLV filter list to the device sensor output.<br>• **exclude**—Specifies the TLVs that must be excluded from the device sensor output.<br>• **include**—Specifies the TLVs that must be included from the device sensor output.<br>• **all**—Disables all notifications for the associated protocol.<br>• **list** *list-name*—Specifies the protocol TLV filter list name. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Tracking TLV Changes

Perform this task to enable client notifications and accounting events for all TLV changes. By default, for each supported peer protocol, client notifications and accounting events will only be generated where an incoming packet includes a TLV that has not previously been received in the context of a given session.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **device-sensor notify all-changes**<br>**Example:**<br>`Device(config)# device-sensor notify all-changes` | Enables client notifications and accounting events for all TLV changes, that is, where either a new TLV is received or a previously received TLV is received with a new value in the context of a given session.<br>**Note**<br>Use the **default device-sensor notify** or the **device-sensor notify new-tlvs** command to return to the default TLV. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Verifying the Device Sensor Configuration

Perform this task to verify the sensor cache entries for all devices.

### Procedure

**Step 1** **enable**

Enables privileged EXEC mode.

**Example:**

```
Device> enable
```

**Step 2** **show device-sensor details**

Displays protocol configuration details for all devices.

**Example:**

```
Device# show device-sensor details
```

**Step 3** **show device-sensor cache mac** *mac-address*

Displays sensor cache entries (the list of protocol TLVs or options received from a device) for a specific device.

**Example:**

```
Device# show device-sensor cache mac 0024.14dc.df4d
```

**Step 4** **show device-sensor cache all**

Displays sensor cache entries for all devices.

**Example:**

```
Device# show device-sensor cache all

Device: 001c.0f74.8480 on port GigabitEthernet2/1
```

# Configuration Examples for Device Sensor

## Examples: Configuring the Device Sensor

The following example shows how to create a Cisco Discovery Protocol filter containing a list of TLVs:

```
Device> enable
Device# configure terminal
Device(config)# device-sensor filter-list cdp list cdp-list
Device(config-sensor-cdplist)# tlv name address-type
Device(config-sensor-cdplist)# tlv name device-name
Device(config-sensor-cdplist)# tlv number 34
Device(config-sensor-cdplist)# end
```

The following example shows how to create an LLDP filter containing a list of TLVs:

```
Device> enable
Device# configure terminal
Device(config)# device-sensor filter-list lldp list lldp-list
Device(config-sensor-lldplist)# tlv name chassis-id
Device(config-sensor-lldplist)# tlv name management-address
Device(config-sensor-lldplist)# tlv number 28
Device(config-sensor-lldplist)# end
```

The following example shows how to create a DHCP filter containing a list of options:

```
Device> enable
Device# configure terminal
Device(config)# device-sensor filter-list dhcp list dhcp-list
Device(config-sensor-lldplist)# option name address-type
Device(config-sensor-lldplist)# option name device-name
Device(config-sensor-lldplist)# option number 34
Device(config-sensor-lldplist)# end
```

The following example shows how to apply a Cisco Discovery Protocol TLV filter list to the device sensor output:

```
Device> enable
Device# configure terminal
Device(config)# device-sensor filter-spec cdp include cdp-list1
Device(config
Device(config-sensor-lldplist)# end)# end
```

The following example shows how to enable client notifications and accounting events for all TLV changes:

```
Device> enable
Device# configure terminal
Device(config)# device-sensor notify all-changes
Device(config)# end
```

# Feature History for Device Sensor

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

***Table 9: Feature Information for Device Sensor***

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Fuji 16.8.1a | Device Sensor | The Device Sensor feature is used to gather raw endpoint data from network devices using protocols such as Cisco Discovery Protocol, Link Layer Discovery Protocol (LLDP), and DHCP. The endpoint data that is gathered is made available to registered clients in the context of an access session. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 10

# Configuring RADIUS


- Prerequisites for Configuring RADIUS, on page 151
- Restrictions for Configuring RADIUS, on page 152
- Information about RADIUS, on page 152
- How to Configure RADIUS, on page 174
- Monitoring CoA Functionality, on page 189
- Feature History for RADIUS, on page 189


## Prerequisites for Configuring RADIUS

This section lists the prerequisites for controlling device access with RADIUS.

General:

- RADIUS and Authentication, Authorization, and Accounting (AAA) must be enabled to use any of the configuration commands in this chapter.
- RADIUS is facilitated through AAA and can be enabled only through AAA commands.
- Use the **aaa new-model** global configuration command to enable AAA.
- Use the **aaa authentication** global configuration command to define method lists for RADIUS authentication.
- Use **line** and **interface** commands to enable the defined method lists to be used.
- At a minimum, you must identify the host or hosts that run the RADIUS server software and define the method lists for RADIUS authentication. You can optionally define method lists for RADIUS authorization and accounting.
- You should have access to and should configure a RADIUS server before configuring RADIUS features on your device.
- The RADIUS host is normally a multiuser system running RADIUS server software from Cisco (Cisco Secure Access Control Server Version 3.0), Livingston, Merit, Microsoft, or another software provider. For more information, see the RADIUS server documentation.
- To use the Change-of-Authorization (CoA) interface, a session must already exist on the switch. CoA can be used to identify a session and enforce a disconnect request. The update affects only the specified session.

RADIUS operation:

- Users must first successfully complete RADIUS authentication before proceeding to RADIUS authorization, if it is enabled.
- For RADIUS over IPv6 configurations, users must enable IPv6 unicast routing by enabling the **ipv6 unicast-routing** command.

# Restrictions for Configuring RADIUS

General:

- To prevent a lapse in security, you cannot configure RADIUS through a network management application.

RADIUS is not suitable in the following network security situations:

- Multiprotocol access environments. RADIUS does not support AppleTalk Remote Access (ARA), NetBIOS Frame Control Protocol (NBFCP), NetWare Asynchronous Services Interface (NASI), or X.25 PAD connections.
- Switch-to-switch or router-to-router situations. RADIUS does not provide two-way authentication. RADIUS can be used to authenticate from one device to a non-Cisco device if the non-Cisco device requires authentication.
- Networks using a variety of services. RADIUS generally binds a user to one service model.

DSCP marking support for RADIUS packets:

- DSCP marking for authentication and accounting is not supported for private servers, fully qualified domain name (FQDN) servers and radsec servers.
- In the case of wired IEEE 802.1x authentication, when source port extension is not enabled, the default ports are in use. The DSCP marking is set to the default ports and all the requests will be marked with the same DSCP value.
- DSCP marking is not supported in the case of wireless IEEE 802.1x authentication, where the source port extension is enabled by default.

# Information about RADIUS

## RADIUS and Switch Access

This section describes how to enable and configure RADIUS. RADIUS provides detailed accounting information and flexible administrative control over the authentication and authorization processes.

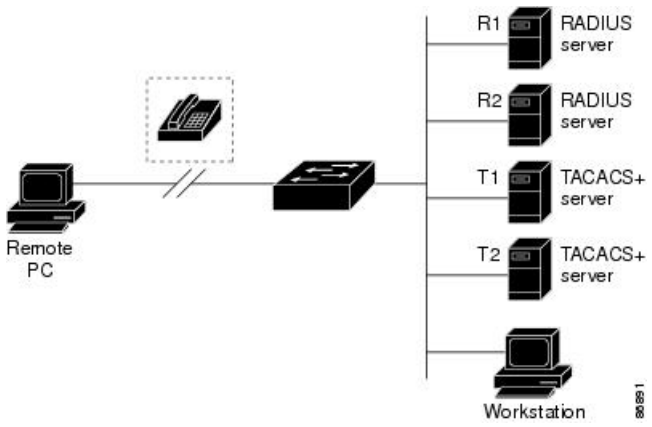
## RADIUS Overview

RADIUS is a distributed client/server system that secures networks against unauthorized access. RADIUS clients run on supported Cisco devices. Clients send authentication requests to a central RADIUS server, which contains all user authentication and network service access information.

Use RADIUS in these network environments that require access security:

- Networks with multiple-vendor access servers, each supporting RADIUS. For example, access servers from several vendors use a single RADIUS server-based security database. In an IP-based network with multiple vendors' access servers, dial-in users are authenticated through a RADIUS server that has been customized to work with the Kerberos security system.
- Turnkey network security environments in which applications support the RADIUS protocol, such as in an access environment that uses a *smart card* access control system.
- Networks already using RADIUS. You can add a Cisco device containing a RADIUS client to the network. This might be the first step when you make a transition to a TACACS+ server. See the illustration: Transitioning from RADIUS to TACACS+ Services below.

***Figure 10: Transitioning from RADIUS to TACACS+ Services***




- Network in which the user must only access a single service. Using RADIUS, you can control user access to a single host, to a single utility such as Telnet, or to the network through a protocol such as IEEE 802.1x. For more information about this protocol, see the chapter *Configuring IEEE 802.1x Port-Based Authentication*.
- Networks that require resource accounting. You can use RADIUS accounting independently of RADIUS authentication or authorization. The RADIUS accounting functions allow data to be sent at the start and end of services, showing the amount of resources (such as time, packets, bytes, and so forth) used during the session. An Internet service provider might use a freeware-based version of RADIUS access control and accounting software to meet special security and billing needs.

# RADIUS Operation

When a user attempts to log in and authenticate to a device that is access controlled by a RADIUS server, these events occur:

1. The user is prompted to enter a username and password.
2. The username and encrypted password are sent over the network to the RADIUS server.
3. The user receives one of the following responses from the RADIUS server:
    - ACCEPT—The user is authenticated.

- REJECT—The user is either not authenticated and is prompted to re-enter the username and password, or access is denied.
- CHALLENGE—A challenge requires additional data from the user.
- CHALLENGE PASSWORD—A response requests the user to select a new password.

The ACCEPT or REJECT response is bundled with additional data that is used for privileged EXEC or network authorization. The additional data included with the ACCEPT or REJECT packets includes these items:

- Telnet, SSH, rlogin, or privileged EXEC services
- Connection parameters, including the host or client IP address, access list, and user timeouts

# RADIUS Change of Authorization

The RADIUS Change of Authorization (CoA) provides a mechanism to change the attributes of an authentication, authorization, and accounting (AAA) session after it is authenticated. When a policy changes for a user or user group in AAA, administrators can send RADIUS CoA packets from the AAA server such as a Cisco Secure Access Control Server (ACS) to reinitialize authentication and apply the new policy. This section provides an overview of the RADIUS interface including available primitives and how they are used during a CoA.

- Change-of-Authorization Requests
- CoA Request Response Code
- CoA Request Commands
- Session Reauthentication
- Stacking Guidelines for Session Termination

A standard RADIUS interface is typically used in a pulled model where the request originates from a network attached device and the response come from the queried servers. Cisco devices support the RADIUS CoA extensions defined in RFC 5176 that are typically used in a pushed model and allow for the dynamic reconfiguring of sessions from external AAA or policy servers.

Cisco devices supports these per-session CoA requests:

- Session reauthentication
- Session termination
- Session termination with port shutdown
- Session termination with port bounce

This feature is integrated with Cisco Secure Access Control Server (ACS) 5.1.

The RADIUS interface is enabled by default on Cisco devices. However, some basic configuration is required for the following attributes:

- Security and Password—refer to the “Preventing Unauthorized Access to Your Switch” section in this guide.

- Accounting—refer to the “Starting RADIUS Accounting” section in the Configuring Switch-Based Authentication chapter in this guide.

Cisco IOS XE software supports the RADIUS CoA extensions defined in RFC 5176 that are typically used in a push model to allow the dynamic reconfiguring of sessions from external AAA or policy servers. Per-session CoA requests are supported for session identification, session termination, host reauthentication, port shutdown, and port bounce. This model comprises one request (CoA-Request) and two possible response codes:

- CoA acknowledgement (ACK) [CoA-ACK]
- CoA nonacknowledgement (NAK) [CoA-NAK]

The request is initiated from a CoA client (typically a AAA or policy server) and directed to the device that acts as a listener.

The table below shows the RADIUS CoA commands and vendor-specific attributes (VSAs) supported by Identity-Based Networking Services. All CoA commands must include the session identifier between the device and the CoA client.

***Table 10: RADIUS CoA Commands Supported by Identity-Based Networking Services***

| CoA Command | Cisco VSA |
|---|---|
| Activate service | Cisco:Avpair=“subscriber:command=activate-service”<br>Cisco:Avpair=“subscriber:service-name=<*service-name*>”<br>Cisco:Avpair=“subscriber:precedence=<*precedence-number*>”<br>Cisco:Avpair=“subscriber:activation-mode=replace-all” |
| Deactivate service | Cisco:Avpair=“subscriber:command=deactivate-service”<br>Cisco:Avpair=“subscriber:service-name=<*service-name*>” |
| Bounce host port | Cisco:Avpair=“subscriber:command=bounce-host-port” |
| Disable host port | Cisco:Avpair=“subscriber:command=disable-host-port” |
| Session query | Cisco:Avpair=“subscriber:command=session-query” |
| Session reauthenticate | Cisco:Avpair=“subscriber:command=reauthenticate”<br>Cisco:Avpair=“subscriber:reauthenticate-type=last” or<br>Cisco:Avpair=“subscriber:reauthenticate-type=rerun” |
| Session terminate | This is a standard disconnect request and does not require a VSA. |
| Interface template | Cisco:AVpair="interface-template-name=<*interfacetemplate*>" |

## Change-of-Authorization Requests

Change of Authorization (CoA) requests, as described in RFC 5176, are used in a push model to allow for session identification, host reauthentication, and session termination. The model is comprised of one request (CoA-Request) and two possible response codes:

- CoA acknowledgment (ACK) [CoA-ACK]

- CoA non-acknowledgment (NAK) [CoA-NAK]

The request is initiated from a CoA client (typically a RADIUS or policy server) and directed to the switch that acts as a listener.

## RFC 5176 Compliance

The Disconnect Request message, which is also referred to as Packet of Disconnect (POD), is supported by the switch for session termination.

This table shows the IETF attributes are supported for this feature.

*Table 11: Supported IETF Attributes*

| Attribute Number | Attribute Name |
|---|---|
| 24 | State |
| 31 | Calling-Station-ID |
| 44 | Acct-Session-ID |
| 80 | Message-Authenticator |
| 101 | Error-Cause |

This table shows the possible values for the Error-Cause attribute.

*Table 12: Error-Cause Values*

| Value | Explanation |
|---|---|
| 201 | Residual Session Context Removed |
| 202 | Invalid EAP Packet (Ignored) |
| 401 | Unsupported Attribute |
| 402 | Missing Attribute |
| 403 | NAS Identification Mismatch |
| 404 | Invalid Request |
| 405 | Unsupported Service |
| 406 | Unsupported Extension |
| 407 | Invalid Attribute Value |
| 501 | Administratively Prohibited |
| 502 | Request Not Routable (Proxy) |
| 503 | Session Context Not Found |

| Value | Explanation |
|---|---|
| 504 | Session Context Not Removable |
| 505 | Other Proxy Processing Error |
| 506 | Resources Unavailable |
| 507 | Request Initiated |
| 508 | Multiple Session Selection Unsupported |

## CoA Request Response Code

The CoA Request response code can be used to convey a command to the switch.

The packet format for a CoA Request Response code as defined in RFC 5176 consists of the following fields: Code, Identifier, Length, Authenticator, and Attributes in the Type:Length:Value (TLV) format. The Attributes field is used to carry Cisco vendor-specific attributes (VSAs).

### Session Identification

For disconnect and CoA requests targeted at a particular session, the switch locates the session based on one or more of the following attributes:

- Acct-Session-Id (IETF attribute #44)
- Audit-Session-Id (Cisco VSA)
- Calling-Station-Id (IETF attribute #31 which contains the host MAC address)
- IPv6 Attributes, which can be one of the following:
  - Framed-IPv6-Prefix (IETF attribute #97) and Framed-Interface-Id (IETF attribute #96), which together create a full IPv6 address per RFC 3162
  - Framed-IPv6-Address
- Plain IP Address (IETF attribute #8)

Unless all session identification attributes included in the CoA message match the session, the switch returns a Disconnect-NAK or CoA-NAK with the “Invalid Attribute Value” error-code attribute.

If more than one session identification attribute is included in the message, all the attributes must match the session or the switch returns a Disconnect- negative acknowledgment (NAK) or CoA-NAK with the error code “Invalid Attribute Value.”

The packet format for a CoA Request code as defined in RFC 5176 consists of the fields: Code, Identifier, Length, Authenticator, and Attributes in Type:Length:Value (TLV) format.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Code      |  Identifier   |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
```

```
|                        Authenticator                        |
|                                                             |
|                                                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Attributes ...
+-+-+-+-+-+-+-+-+-+-+-+-+-
```

The attributes field is used to carry Cisco vendor-specific attributes (VSAs).

For CoA requests targeted at a particular enforcement policy, the device returns a CoA-NAK with the error code “Invalid Attribute Value” if any of the above session identification attributes are included in the message.

### CoA ACK Response Code

If the authorization state is changed successfully, a positive acknowledgment (ACK) is sent. The attributes returned within CoA ACK will vary based on the CoA Request and are discussed in individual CoA Commands.

### CoA NAK Response Code

A negative acknowledgment (NAK) indicates a failure to change the authorization state and can include attributes that indicate the reason for the failure. Use **show** commands to verify a successful CoA.

## CoA Request Commands

***Table 13: Supported CoA Commands***

| Command [1] | Cisco VSA |
|---|---|
| Reauthenticate host | Cisco:Avpair=“subscriber:command=reauthenticate” |
| Terminate session | This is a standard disconnect request that does not require a VSA. |
| Bounce host port | Cisco:Avpair=“subscriber:command=bounce-host-port” |
| Disable host port | Cisco:Avpair=“subscriber:command=disable-host-port” |

[1] All CoA commands must include the session identifier between the device and the CoA client.

### Session Reauthentication

The AAA server typically generates a session reauthentication request when a host with an unknown identity or posture joins the network and is associated with a restricted access authorization profile (such as a guest VLAN). A reauthentication request allows the host to be placed in the appropriate authorization group when its credentials are known.

To initiate session authentication, the AAA server sends a standard CoA-Request message which contains a Cisco VSA in this form: *Cisco:Avpair=“subscriber:command=reauthenticate”* and one or more session identification attributes.

The current session state determines the switch response to the message. If the session is currently authenticated by IEEE 802.1x, the switch responds by sending an EAPoL (Extensible Authentication Protocol over Lan) -RequestId message to the server.

If the session is currently authenticated by MAC authentication bypass (MAB), the switch sends an access-request to the server, passing the same identity attributes used for the initial successful authentication.

If session authentication is in progress when the switch receives the command, the switch terminates the process, and restarts the authentication sequence, starting with the method configured to be attempted first.

If the session is not yet authorized, or is authorized via guest VLAN, or critical VLAN, or similar policies, the reauthentication message restarts the access control methods, beginning with the method configured to be attempted first. The current authorization of the session is maintained until the reauthentication leads to a different authorization result.

### Session Termination

There are three types of CoA requests that can trigger session termination. A CoA Disconnect-Request terminates the session, without disabling the host port. This command causes re-initialization of the authenticator state machine for the specified host, but does not restrict that host access to the network.

To restrict a host's access to the network, use a CoA Request with the Cisco:Avpair="subscriber:command=disable-host-port" VSA. This command is useful when a host is known to be causing problems on the network, and you need to immediately block network access for the host. When you want to restore network access on the port, re-enable it using a non-RADIUS mechanism.

When a device with no supplicant, such as a printer, needs to acquire a new IP address (for example, after a VLAN change), terminate the session on the host port with port-bounce (temporarily disable and then re-enable the port).

### CoA Disconnect-Request

This command is a standard Disconnect-Request. If the session cannot be located, the device returns a Disconnect-NAK message with the "Session Context Not Found" error-code attribute. If the session is located, the device terminates the session. After the session has been completely removed, the device returns a Disconnect-ACK.

If the device fails-over to a standby device before returning a Disconnect-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the session is not found following re-sending, a Disconnect-ACK is sent with the "Session Context Not Found" error-code attribute.

### CoA Request: Disable Host Port

The RADIUS server CoA disable port command administratively shuts down the authentication port that is hosting a session, resulting in session termination. This command is useful when a host is known to cause problems on the network and network access needs to be immediately blocked for the host. To restore network access on the port, reenable it using a non-RADIUS mechanism. This command is carried in a standard CoA-Request message that has this new vendor-specific attribute (VSA):

Cisco:Avpair="subscriber:command=disable-host-port"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the "Session Identification" section. If the session cannot be located, the device returns a CoA-NAK message with the "Session Context Not Found" error-code attribute. If the session is located, the device disables the hosting port and returns a CoA-ACK message.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation has completed, the operation is restarted on the new active device.

**Note** A Disconnect-Request failure following command re-sending could be the result of either a successful session termination before change-over (if the Disconnect-ACK was not sent) or a session termination by other means (for example, a link failure) that occurred after the original command was issued and before the standby device became active.

### CoA Request: Bounce-Port

A RADIUS server CoA bounce port sent from a RADIUS server can cause a link flap on an authentication port, which triggers DHCP renegotiation from one or more hosts connected to this port. This incident can occur when there is a VLAN change and the endpoint is a device (such as a printer) that does not have a mechanism to detect a change on this authentication port. The CoA bounce port is carried in a standard CoA-Request message that contains the following VSA:

Cisco:Avpair="subscriber:command=bounce-host-port"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes. If the session cannot be located, the device returns a CoA-NAK message with the “Session Context Not Found” error-code attribute. If the session is located, the device disables the hosting port for a period of 10 seconds, re-enables it (port-bounce), and returns a CoA-ACK.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation has completed, the operation is re-started on the new active device.

## Default RADIUS Configuration

RADIUS and AAA are disabled by default.

To prevent a lapse in security, you cannot configure RADIUS through a network management application. When enabled, RADIUS can authenticate users accessing the device through the CLI.

## RADIUS Server Host

Device-to-RADIUS-server communication involves several components:

- Hostname or IP address
- Authentication destination port
- Accounting destination port
- Key string
- Timeout period
- Retransmission value

You identify RADIUS security servers by their hostname or IP address, hostname and specific UDP port numbers, or their IP address and specific UDP port numbers. The combination of the IP address and the UDP port number creates a unique identifier, allowing different ports to be individually defined as RADIUS hosts providing a specific AAA service. This unique identifier enables RADIUS requests to be sent to multiple UDP ports on a server at the same IP address.

If two different host entries on the same RADIUS server are configured for the same service—for example, accounting—the second host entry configured acts as a fail-over backup to the first one. Using this example, if the first host entry fails to provide accounting services, the %RADIUS-4-RADIUS_DEAD message appears, and then the device tries the second host entry configured on the same device for accounting services. (The RADIUS host entries are tried in the order that they are configured.)

A RADIUS server and the device use a shared secret text string to encrypt passwords and exchange responses. To configure RADIUS to use the AAA security commands, you must specify the host running the RADIUS server daemon and a secret text (key) string that it shares with the device.

The timeout, retransmission, and encryption key values can be configured globally for all RADIUS servers, on a per-server basis, or in some combination of global and per-server settings.

# RADIUS Login Authentication

To configure AAA authentication, you define a named list of authentication methods and then apply that list to various ports. The method list defines the types of authentication to be performed and the sequence in which they are performed; it must be applied to a specific port before any of the defined authentication methods are performed. The only exception is the default method list. The default method list is automatically applied to all ports except those that have a named method list explicitly defined.

A method list describes the sequence and authentication methods to be queried to authenticate a user. You can designate one or more security protocols to be used for authentication, thus ensuring a backup system for authentication in case the initial method fails. The software uses the first method listed to authenticate users; if that method fails to respond, the software selects the next authentication method in the method list. This process continues until there is successful communication with a listed authentication method or until all defined methods are exhausted. If authentication fails at any point in this cycle—meaning that the security server or local username database responds by denying the user access—the authentication process stops, and no other authentication methods are attempted.

# AAA Server Groups

You can configure the device to use AAA server groups to group existing server hosts for authentication. You select a subset of the configured server hosts and use them for a particular service. The server group is used with a global server-host list, which lists the IP addresses of the selected server hosts.

Server groups also can include multiple host entries for the same server if each entry has a unique identifier (the combination of the IP address and UDP port number), allowing different ports to be individually defined as RADIUS hosts providing a specific AAA service. This unique identifier enables RADIUS requests to be sent to different UDP ports on a server at the same IP address. If you configure two different host entries on the same RADIUS server for the same service, (for example, accounting), the second configured host entry acts as a fail-over backup to the first one. If the first host entry fails to provide accounting services, the network access server tries the second host entry configured on the same device for accounting services. (The RADIUS host entries are tried in the order in which they are configured.)

# AAA Authorization

AAA authorization limits the services available to a user. When AAA authorization is enabled, the device uses information retrieved from the user's profile, which is in the local user database or on the security server, to configure the user's session. The user is granted access to a requested service only if the information in the user profile allows it.

# RADIUS Accounting

The AAA accounting feature tracks the services that users are using and the amount of network resources that they are consuming. When you enable AAA accounting, the device reports user activity to the RADIUS security server in the form of accounting records. Each accounting record contains accounting attribute-value (AV) pairs and is stored on the security server. You can then analyze the data for network management, client billing, or auditing.

# Vendor-Specific RADIUS Attributes

The Internet Engineering Task Force (IETF) draft standard specifies a method for communicating vendor-specific information between the device and the RADIUS server by using the vendor-specific attribute (attribute 26). Vendor-specific attributes (VSAs) allow vendors to support their own extended attributes not suitable for general use. The Cisco RADIUS implementation supports one vendor-specific option by using the format recommended in the specification. Cisco's vendor-ID is 9, and the supported option has vendor-type 1, which is named *cisco-avpair*. The value is a string with this format:

```
protocol : attribute sep value *
```

*Protocol* is a value of the Cisco protocol attribute for a particular type of authorization. *Attribute* and *value* are an appropriate attributevalue (AV) pair defined in the Cisco TACACS+ specification, and *sep* is = for mandatory attributes and is * for optional attributes. The full set of features available for TACACS+ authorization can then be used for RADIUS.

For example, the following AV pair causes Cisco's "multiple named IP address pools" feature to be activated during IP authorization (during PPP's Internet Protocol Control Protocol (IPCP) address assignment):

```
cisco-avpair= ″ip:addr-pool=first‟
```

If you insert an "*", the AV pair "ip:addr-pool=first" becomes optional. Note that any AV pair can be made optional:

```
cisco-avpair= ″ip:addr-pool*first‟
```

The following example shows how to cause a user logging in from a network access server to have immediate access to EXEC commands:

```
cisco-avpair= ″shell:priv-lvl=15‟
```

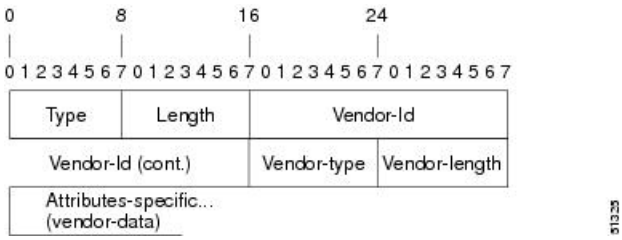
Other vendors have their own unique vendor-IDs, options, and associated VSAs. For more information about vendor-IDs and VSAs, see RFC 2138, "Remote Authentication Dial-In User Service (RADIUS)."

Attribute 26 contains the following three elements:

- Type
- Length
- String (also known as data)
  - Vendor-ID
  - Vendor-Type
  - Vendor-Length
  - Vendor-Data

The figure below shows the packet format for a VSA encapsulated "behind" attribute 26.

*Figure 11: VSA Encapsulated Behind Attribute 26*




**Note** It is up to the vendor to specify the format of their VSA. The Attribute-Specific field (also known as Vendor-Data) is dependent on the vendor's definition of that attribute.

The table below describes significant fields listed in the Vendor-Specific RADIUS IETF Attributes table (second table below), which lists supported vendor-specific RADIUS attributes (IETF attribute 26).

*Table 14: Vendor-Specific Attributes Table Field Descriptions*

| Field | Description |
|---|---|
| Number | All attributes listed in the following table are extensions of IETF attribute 26. |
| Vendor-Specific Command Codes | A defined code used to identify a particular vendor. Code 9 defines Cisco VSAs, 311 defines Microsoft VSAs, and 529 defines Ascend VSAs. |
| Sub-Type Number | The attribute ID number. This number is much like the ID numbers of IETF attributes, except it is a “second layer” ID number encapsulated behind attribute 26. |
| Attribute | The ASCII string name of the attribute. |
| Description | Description of the attribute. |

*Table 15: Vendor-Specific RADIUS IETF Attributes*

| Number | Vendor-Specific Company Code | Sub-Type Number | Attribute | Description |
|---|---|---|---|---|
| MS-CHAP Attributes | | | | |
| 26 | 311 | 1 | MSCHAP-Response | Contains the response value provided by a PPP MS-CHAP user in response to the challenge. It is only used in Access-Request packets. This attribute is identical to the PPP CHAP Identifier. ( RFC 2548 |

| Number | Vendor-Specific Company Code | Sub-Type Number | Attribute | Description |
|---|---|---|---|---|
| 26 | 311 | 11 | MSCHAP-Challenge | Contains the challenge sent by a network access server to an MS-CHAP user. It can be used in both Access-Request and Access-Challenge packets. ( RFC 2548 ) |
| VPDN Attributes | | | | |
| 26 | 9 | 1 | l2tp-cm-local-window-size | Specifies the maximum receive window size for L2TP control messages. This value is advertised to the peer during tunnel establishment. |
| 26 | 9 | 1 | l2tp-drop-out-of-order | Respects sequence numbers on data packets by dropping those that are received out of order. This does not ensure that sequence numbers will be sent on data packets, just how to handle them if they are received. |
| 26 | 9 | 1 | l2tp-hello-interval | Specifies the number of seconds for the hello keepalive interval. Hello packets are sent when no data has been sent on a tunnel for the number of seconds configured here. |
| 26 | 9 | 1 | l2tp-hidden-avp | When enabled, sensitive AVPs in L2TP control messages are scrambled or hidden. |
| 26 | 9 | 1 | l2tp-nosession-timeout | Specifies the number of seconds that a tunnel will stay active with no sessions before timing out and shutting down. |

| Number | Vendor-Specific Company Code | Sub-Type Number | Attribute | Description |
|---|---|---|---|---|
| 26 | 9 | 1 | tunnel-tos-reflect | Copies the IP ToS field from the IP header of each payload packet to the IP header of the tunnel packet for packets entering the tunnel at the LNS. |
| 26 | 9 | 1 | l2tp-tunnel-authen | If this attribute is set, it performs L2TP tunnel authentication. |
| 26 | 9 | 1 | l2tp-tunnel-password | Shared secret used for L2TP tunnel authentication and AVP hiding. |
| 26 | 9 | 1 | l2tp-udp-checksum | This is an authorization attribute and defines whether L2TP should perform UDP checksums for data packets. Valid values are “yes” and “no.” The default is no. |
| Store and Forward Fax Attributes | | | | |
| 26 | 9 | 3 | Fax-Account-Id-Origin | Indicates the account ID origin as defined by system administrator for the **mmoip aaa receive-id** or the **mmoip aaa send-id** commands. |
| 26 | 9 | 4 | Fax-Msg-Id= | Indicates a unique fax message identification number assigned by Store and Forward Fax. |
| 26 | 9 | 5 | Fax-Pages | Indicates the number of pages transmitted or received during this fax session. This page count includes cover pages. |

| Number | Vendor-Specific Company Code | Sub-Type Number | Attribute | Description |
|---|---|---|---|---|
| 26 | 9 | 6 | Fax-Coverpage-Flag | Indicates whether or not a cover page was generated by the off-ramp gateway for this fax session. True indicates that a cover page was generated; false means that a cover page was not generated. |
| 26 | 9 | 7 | Fax-Modem-Time | Indicates the amount of time in seconds the modem sent fax data (x) and the amount of time in seconds of the total fax session (y), which includes both fax-mail and PSTN time, in the form x/y. For example, 10/15 means that the transfer time took 10 seconds, and the total fax session took 15 seconds. |
| 26 | 9 | 8 | Fax-Connect-Speed | Indicates the modem speed at which this fax-mail was initially transmitted or received. Possible values are 1200, 4800, 9600, and 14400. |
| 26 | 9 | 9 | Fax-Recipient-Count | Indicates the number of recipients for this fax transmission. Until e-mail servers support Session mode, the number should be 1. |
| 26 | 9 | 10 | Fax-Process-Abort-Flag | Indicates that the fax session was terminated or successful. True means that the session was terminated; false means that the session was successful. |
| 26 | 9 | 11 | Fax-Dsn-Address | Indicates the address to which DSNs will be sent. |

| Number | Vendor-Specific Company Code | Sub-Type Number | Attribute | Description |
|---|---|---|---|---|
| 26 | 9 | 12 | Fax-Dsn-Flag | Indicates whether or not DSN has been enabled. True indicates that DSN has been enabled; false means that DSN has not been enabled. |
| 26 | 9 | 13 | Fax-Mdn-Address | Indicates the address to which MDNs will be sent. |
| 26 | 9 | 14 | Fax-Mdn-Flag | Indicates whether or not message delivery notification (MDN) has been enabled. True indicates that MDN had been enabled; false means that MDN had not been enabled. |
| 26 | 9 | 15 | Fax-Auth-Status | Indicates whether or not authentication for this fax session was successful. Possible values for this field are success, failed, bypassed, or unknown. |
| 26 | 9 | 16 | Email-Server-Address | Indicates the IP address of the e-mail server handling the on-ramp fax-mail message. |
| 26 | 9 | 17 | Email-Server-Ack-Flag | Indicates that the on-ramp gateway has received a positive acknowledgment from the e-mail server accepting the fax-mail message. |
| 26 | 9 | 18 | Gateway-Id | Indicates the name of the gateway that processed the fax session. The name appears in the following format: hostname.domain-name. |
| 26 | 9 | 19 | Call-Type | Describes the type of fax activity: fax receive or fax send. |

| Number | Vendor-Specific Company Code | Sub-Type Number | Attribute | Description |
|---|---|---|---|---|
| 26 | 9 | 20 | Port-Used | Indicates the slot/port number of the Cisco AS5300 used to either transmit or receive this fax-mail. |
| 26 | 9 | 21 | Abort-Cause | If the fax session terminates, indicates the system component that signaled the termination. Examples of system components that could trigger an termination are FAP (Fax Application Process), TIFF (the TIFF reader or the TIFF writer), fax-mail client, fax-mail server, ESMTP client, or ESMTP server. |
| H323 Attributes | | | | |
| 26 | 9 | 23 | Remote-Gateway-ID (h323-remote-address) | Indicates the IP address of the remote gateway. |
| 26 | 9 | 24 | Connection-ID (h323-conf-id) | Identifies the conference ID. |
| 26 | 9 | 25 | Setup-Time (h323-setup-time) | Indicates the setup time for this connection in Coordinated Universal Time (UTC) formerly known as Greenwich Mean Time (GMT) and Zulu time. |
| 26 | 9 | 26 | Call-Origin (h323-call-origin) | Indicates the origin of the call relative to the gateway. Possible values are originating and terminating (answer). |
| 26 | 9 | 27 | Call-Type (h323-call-type) | Indicates call leg type. Possible values are **telephony** and **VoIP**. |
| 26 | 9 | 28 | Connect-Time (h323-connect-time) | Indicates the connection time for this call leg in UTC. |

| Number | Vendor-Specific Company Code | Sub-Type Number | Attribute | Description |
|---|---|---|---|---|
| 26 | 9 | 29 | Disconnect-Time (h323-disconnect-time) | Indicates the time this call leg was disconnected in UTC. |
| 26 | 9 | 30 | Disconnect-Cause (h323-disconnect-cause) | Specifies the reason a connection was taken offline per Q.931 specification. |
| 26 | 9 | 31 | Voice-Quality (h323-voice-quality) | Specifies the impairment factor (ICPIF) affecting voice quality for a call. |
| 26 | 9 | 33 | Gateway-ID (h323-gw-id) | Indicates the name of the underlying gateway. |
| Large Scale Dialout Attributes | | | | |
| 26 | 9 | 1 | callback-dialstring | Defines a dialing string to be used for callback. |
| 26 | 9 | 1 | data-service | No description available. |
| 26 | 9 | 1 | dial-number | Defines the number to dial. |
| 26 | 9 | 1 | force-56 | Determines whether the network access server uses only the 56 K portion of a channel, even when all 64 K appear to be available. |
| 26 | 9 | 1 | map-class | Allows the user profile to reference information configured in a map class of the same name on the network access server that dials out. |
| 26 | 9 | 1 | send-auth | Defines the protocol to use (PAP or CHAP) for username-password authentication following CLID authentication. |

| Number | Vendor-Specific Company Code | Sub-Type Number | Attribute | Description |
|---|---|---|---|---|
| 26 | 9 | 1 | send-name | PPP name authentication. To apply for PAP, do not configure the **ppp pap sent-name password** command on the interface. For PAP, “preauth:send-name” and “preauth:send-secret” will be used as the PAP username and PAP password for outbound authentication. For CHAP, “preauth:send-name” will be used not only for outbound authentication, but also for inbound authentication. For a CHAP inbound case, the NAS will use the name defined in “preauth:send-name” in the challenge packet to the caller box.<br>**Note**<br>The send-name attribute has changed over time: Initially, it performed the functions now provided by both the send-name and remote-name attributes. Because the remote-name attribute has been added, the send-name attribute is restricted to its current behavior. |

| Number | Vendor-Specific Company Code | Sub-Type Number | Attribute | Description |
|---|---|---|---|---|
| 26 | 9 | 1 | send-secret | PPP password authentication. The vendor-specific attributes (VSAs) “preauth:send-name” and “preauth:send-secret” will be used as the PAP username and PAP password for outbound authentication. For a CHAP outbound case, both “preauth:send-name” and “preauth:send-secret” will be used in the response packet. |
| 26 | 9 | 1 | remote-name | Provides the name of the remote host for use in large-scale dial-out. Dialer checks that the large-scale dial-out remote name matches the authenticated name, to protect against accidental user RADIUS misconfiguration. (For example, dialing a valid phone number but connecting to the wrong device.) |
| Miscellaneous Attributes | | | | |

| Number | Vendor-Specific Company Code | Sub-Type Number | Attribute | Description |
|---|---|---|---|---|
| 26 | 9 | 2 | Cisco-NAS-Port | Specifies additional vendor specific attribute (VSA) information for NAS-Port accounting. To specify additional NAS-Port information in the form an Attribute-Value Pair (AVPair) string, use the **radius-server vsa send** global configuration command.<br>**Note**<br>This VSA is typically used in Accounting, but may also be used in Authentication (Access-Request) packets. |
| 26 | 9 | 1 | min-links | Sets the minimum number of links for MLP. |
| 26 | 9 | 1 | proxyacl#<n> | Allows users to configure the downloadable user profiles (dynamic ACLs) by using the authentication proxy feature so that users can have the configured authorization to permit traffic going through the configured interfaces. |

| Number | Vendor-Specific Company Code | Sub-Type Number | Attribute | Description |
|---|---|---|---|---|
| 26 | 9 | 1 | spi | Carries the authentication information needed by the home agent to authenticate a mobile node during registration. The information is in the same syntax as the **ip mobile secure host <addr>** configuration command. Basically it contains the rest of the configuration command that follows that string, verbatim. It provides the Security Parameter Index (SPI), key, authentication algorithm, authentication mode, and replay protection timestamp range. |

# Vendor-Proprietary RADIUS Server Communication

Although an IETF draft standard for RADIUS specifies a method for communicating vendor-proprietary information between the device and the RADIUS server, some vendors have extended the RADIUS attribute set in a unique way. Cisco IOS XE software supports a subset of vendor-proprietary RADIUS attributes.

As mentioned earlier, to configure RADIUS (whether vendor-proprietary or IETF draft-compliant), you must specify the host running the RADIUS server daemon and the secret text string it shares with the device. You specify the RADIUS host and secret text string by using the **radius server** global configuration commands.

# DSCP marking for RADIUS packets

Differentiated Services (DiffServ) is a model in which traffic is treated by intermediate systems with relative priorities based on the type of services (ToS) field. The six most significant bits of the DiffServ field is called as the Differentiated Services Code Point (DSCP). Cisco IOS XE software supports DSCP marking for RADIUS packets. DSCP marking enables faster authentication and accounting of RADIUS packets.

You can configure DSCP marking on the RADIUS server, server group and in global configuration mode. When DSCP marking configuration is applied on RADIUS server, server group and global configuration mode, the DSCP marking values entered on the RADIUS server is taken.

- If there is no DSCP marking configuration on the RADIUS server, the DSCP marking values configured on the server group is applied to the RADIUS packets.
- If there is no DSCP marking configuration on the RADIUS server, RADIUS server group, the DSCP marking values configured at the global configuration mode is applied to the RADIUS packets.

# How to Configure RADIUS

## Identifying the RADIUS Server Host

To apply these settings globally to all RADIUS servers communicating with the device, use the three unique global configuration commands: **radius-server timeout**, **radius-server retransmit**, and **key** *string*.

You can configure the device to use AAA server groups to group existing server hosts for authentication.

You also need to configure some settings on the RADIUS server. These settings include the IP address of the device and the key string to be shared by both the server and the device.

Follow these steps to configure per-server RADIUS server communication.

#### Before you begin

If you configure both global and per-server functions (timeout, retransmission, and key commands) on the device, the per-server timer, retransmission, and key value commands override global timer, retransmission, and key value commands.

#### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **radius server** *server name*<br>**Example:**<br>`Device(config)# radius server rsim` | Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode. |
| **Step 4** | **address** {**ipv4** \| **ipv6**}*ip address*{ **auth-port** *port number* \| **acct-port** *port number*}<br>**Example:**<br>`Device(config-radius-server)# address ipv4 124.2.2.12 auth-port 1612` | (Optional) Specifies the RADIUS server parameters.<br>For **auth-port** *port-number*, specify the UDP destination port for authentication requests. The default is 1645. The range is 0 to 65536.<br>For **acct-port** *port-number*, specify the UDP destination port for authentication requests. The default is 1646. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **key** *string*<br>**Example:**<br>`Device(config-radius-server)# key rad123` | (Optional) For **key** *string*, specify the authentication and encryption key used between the device and the RADIUS daemon running on the RADIUS server.<br>**Note**<br>The key is a text string that must match the encryption key used on the RADIUS server. Always configure the key as the last item in the **radius server** command. Leading spaces are ignored, but spaces within and at the end of the key are used. If you use spaces in your key, do not enclose the key in quotation marks unless the quotation marks are part of the key. |
| **Step 6** | **retransmit** *value*<br>**Example:**<br>`Device(config-radius-server)# retransmit 10` | (Optional) Specifies the number of times a RADIUS request is resent when the server is not responding or responding slowly. The range is 1 to 100. This setting overrides the **radius-server retransmit** global configuration command setting. |
| **Step 7** | **timeout** *seconds*<br>**Example:**<br>`Device(config-radius-server)# timeout 60` | (Optional) Specifies the time interval that the device waits for the RADIUS server to reply before sending a request again. The range is 1 to 1000. This setting overrides the **radius-server timeout** global configuration command setting.<br>**Note**<br>We recommend that you configure timeout under the **radius-server timeout** command only and not under the **aaa group server radius** command. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config-radius-server)# end` | Exits RADIUS server configuration mode and enters privileged EXEC mode. |

# Configuring RADIUS Login Authentication

Follow these steps to configure RADIUS login authentication:

### Before you begin

To secure the device for HTTP access by using AAA methods, you must configur the **ip http authentication aaa** global configuration command. Configuring AAA authentication does not secure the device for HTTP access by using AAA methods.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authentication login** {**default** \| *list-name*} *method1* [*method2...*]<br>**Example:**<br>`Device(config)# aaa authentication login default local` | Creates a login authentication method list.<br>• To create a default list that is used when a named list is *not* specified in the **login authentication** command, use the **default** keyword followed by the methods that are to be used in default situations. The default method list is automatically applied to all ports.<br>• For *list-name*, specify a character string to name the list you are creating.<br>• For *method1...*, specify the actual method the authentication algorithm tries. The additional methods of authentication are used only if the previous method returns an error, not if it fails.<br>Select one of these methods:<br>• *enable*—Use the enable password for authentication. Before you can use this authentication method, you must define an enable password by using the **enable** *password* global configuration command. |

| | Command or Action | Purpose |
|---|---|---|
| | | • *group radius*—Use RADIUS authentication. Before you can use this authentication method, you must configure the RADIUS server.<br>• *line*—Use the line password for authentication. Before you can use this authentication method, you must define a line password. Use the **password** *password* line configuration command.<br>• *local*—Use the local username database for authentication. You must enter username information in the database. Use the **username** *name* **password** global configuration command.<br>• *local-case*—Use a case-sensitive local username database for authentication. You must enter username information in the database by using the **username** *password* global configuration command.<br>• *none*—Do not use any authentication for login. |
| **Step 5** | **line** [**console** \| **tty** \| **vty**] *line-number* [*ending-line-number*]<br>**Example:**<br>`Device(config)# line 1 4` | Enters line configuration mode, and configure the lines to which you want to apply the authentication list. |
| **Step 6** | **login authentication** {**default** \| *list-name*}<br>**Example:**<br>`Device(config-line)# login authentication default` | Applies the authentication list to a line or set of lines.<br>• If you specify **default**, use the default list created with the **aaa authentication login** command.<br>• For *list-name*, specify the list created with the **aaa authentication login** command. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-line)# end` | Exits line configuration mode and enters privileged EXEC mode. |

# Defining AAA Server Groups

You use the **server** group server configuration command to associate a particular server with a defined group server. You can either identify the server by its IP address or identify multiple host instances or entries by using the optional **auth-port** and **acct-port** keywords.

Follow these steps to define AAA server groups:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **radius server** *name*<br>**Example:**<br>`Device(config)# radius server ISE` | Specifies the name of the RADIUS server configuration for Protected Access Credential (PAC) provisioning and enters RADIUS server configuration mode.<br>The device also supports RADIUS for IPv6. |
| **Step 4** | **address** {**ipv4** \| **ipv6**} {*ip-address* \| *hostname*} **auth-port** *port-number* **acct-port** *port-number*<br>**Example:**<br>`Device(config-radius-server)# address ipv4 10.1.1.1 auth-port 1645 acct-port 1646` | Configures the IPv4 address for the RADIUS server accounting and authentication parameters. |
| **Step 5** | **key** *string*<br>**Example:**<br>`Device(config-radius-server)# key cisco123` | Specifies the authentication and encryption key for all RADIUS communications between the device and the RADIUS server. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(config-radius-server)# exit` | Exits RADIUS server configuration mode and enters global configuration mode. |
| **Step 7** | **aaa group server radius** *group_name*<br>**Example:**<br>`Device(config)# aaa group server radius abc` | Defines the RADIUS server group configuration and enters RADIUS server group configuration mode. |
| **Step 8** | **server name** *server*<br>**Example:** | Associates the RADIUS server to the server group. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-sg-radius)# server name ISE` | |
| **Step 9** | **end**<br>**Example:**<br>`Device(config-sg-radius)# end` | Exits RADIUS server group configuration mode and returns to privileged EXEC mode. |

# Configuring RADIUS Authorization for User Privileged Access and Network Services

**Note** Authorization is bypassed for authenticated users who log in through the CLI even if authorization has been configured.

Follow these steps to configure RADIUS authorization for user priviledged access and network services:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa authorization network** *authorization-list* **radius**<br>**Example:**<br>`Device(config)# aaa authorization network list1 radius` | Configures the device for user RADIUS authorization for all network-related service requests. |
| **Step 4** | **aaa authorization exec** *authorization-list* **radius**<br>**Example:**<br>`Device(config)# aaa authorization exec list1 radius` | Configures the device for user RADIUS authorization if the user has privileged EXEC access.<br>The **exec** keyword might return user profile information (such as **autocommand** information). |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

### What to do next

You can use the **aaa authorization** global configuration command with the **radius** keyword to set parameters that restrict a user's network access to privileged EXEC mode.

The **aaa authorization exec radius local** command sets these authorization parameters:

- Use RADIUS for privileged EXEC access authorization if authentication was performed by using RADIUS.
- Use the local database if authentication was not performed by using RADIUS.

# Starting RADIUS Accounting

Follow these steps to start RADIUS accounting:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa accounting network** *accounting-list* **start-stop radius**<br>**Example:**<br>`Device(config)# aaa accounting network start-stop radius` | Enables RADIUS accounting for all network-related service requests. |
| **Step 4** | **aaa accounting exec** *accounting-list* **start-stop radius**<br>**Example:**<br>`Device(config)# aaa accounting exec acc-list start-stop radius` | Enables RADIUS accounting to send a start-record accounting notice at the beginning of a privileged EXEC process and a stop-record at the end. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring Settings for All RADIUS Servers

Beginning in privileged EXEC mode, follow these steps to configure settings for all RADIUS servers:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **radius server** *server name*<br>**Example:**<br>`Device(config)# radius server rsim` | Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode. |
| **Step 4** | **key** *string*<br>**Example:**<br>`Device(config-radius-server)# key your_server_key` | Specifies the shared secret text string used between the switch and all RADIUS servers.<br>**Note**<br>The key is a text string that must match the encryption key used on the RADIUS server. Leading spaces are ignored, but spaces within and at the end of the key are used. If you use spaces in your key, do not enclose the key in quotation marks unless the quotation marks are part of the key. |
| **Step 5** | **retransmit** *retries*<br>**Example:**<br>`Device(config-radius-server)# retransmit 5` | Specifies the number of times the switch sends each RADIUS request to the server before giving up. The default is 3; the range 1 to 1000. |
| **Step 6** | **timeout** *seconds*<br>**Example:**<br>`Device(config-radius-server)# timeout 3` | Specifies the number of seconds a switch waits for a reply to a RADIUS request before resending the request. The default is 5 seconds; the range is 1 to 1000. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **end**<br>**Example:**<br>`Device(config-radius-server)# end` | Exits RADIUS server configuration mode and enters privileged EXEC mode. |

# Configuring the Device to Use Vendor-Specific RADIUS Attributes

Follow these steps to configure vendor-specific RADIUS attributes:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **radius-server vsa send** [**accounting** \| **authentication**]<br>**Example:**<br>`Device(config)# radius-server vsa send accounting` | Enables the device to recognize and use VSAs as defined by RADIUS IETF attribute 26.<br>• (Optional) Use the **accounting** keyword to limit the set of recognized vendor-specific attributes to only accounting attributes.<br>• (Optional) Use the **authentication** keyword to limit the set of recognized vendor-specific attributes to only authentication attributes.<br>If you enter this command without keywords, both accounting and authentication vendor-specific attributes are used. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and enters privileged EXEC mode. |

# Configuring the Device for Vendor-Proprietary RADIUS Server Communication

Follow these steps to configure vendor-proprietary RADIUS server communication:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **radius server** *server name*<br>**Example:**<br>`Device(config)# radius server rsim` | Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode. |
| **Step 4** | **address** { **ipv4 \| ipv6** } *ip address*<br>**Example:**<br>`Device(config-radius-server)# address ipv4 172.24.25.10` | (Optional) Specifies the IP address of the RADIUS server. |
| **Step 5** | **non-standard**<br>**Example:**<br>`Device(config-radius-server)# non-standard` | Identifies that the RADIUS server using a vendor-proprietary implementation of RADIUS. |
| **Step 6** | **key** *string*<br>**Example:**<br>`Device(config-radius-server)# key rad123` | Specifies the shared secret text string used between the device and the vendor-proprietary RADIUS server. The device and the RADIUS server use this text string to encrypt passwords and exchange responses. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-radius-server)# end` | Exits RADIUS server mode and enters privileged EXEC mode. |

# Configuring DSCP Marking on a RADIUS Server

Follow these steps to configure DSCP marking for authentication and accounting on a radius server:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **radius server** *server_name*<br>**Example:**<br>`Device(config)# radius server rsim` | Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode. |
| **Step 4** | **address** { **ipv4** \| **ipv6** } *ip address* [**auth-port** *auth_port_number* **acct-port** *acct_port_number*]<br>**Example:**<br>`Device(config-radius-server)# address ipv4 10.1.1.1 auth-port 1645 acct-port 1646` | (Optional) Specifies the IP address of the RADIUS server.<br>• **auth-port** configures the port value for radius authentication server. The default value is 1812.<br>• **acct-port** configures the port value for radius accounting server. The default value is 1813. |
| **Step 5** | **dscp** {**acct** *dscp_acct_value* \| **auth** *dscp_auth_value*}<br>**Example:**<br>`Device(config-radius-server)# dscp auth 10 acct 20` | Configures DSCP marking for authentication and accounting on the radius server.<br>• **acct** configures radius DSCP marking value for accounting. The valid range is from 1 to 63. The default value is 0.<br>• **auth** configures radius DSCP marking value for authentication. The valid range is from 1 to 63. The default value is 0. |
| **Step 6** | **key** *string*<br>**Example:**<br>`Device(config-radius-server)# key rad123` | Specifies the shared secret text string used between the device and the vendor-proprietary RADIUS server. The device and the RADIUS server use this text string to encrypt passwords and exchange responses. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-radius-server)# end` | Exits RADIUS server mode and enters privileged EXEC mode. |

# Configuring the Source Interface and DSCP Marking on RADIUS Server Group

Follow these steps to configure the source interface and DSCP marking for authentication and accounting on radius server groups:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa group server radius** *group_name*<br>**Example:**<br>`Device(config)# aaa group server radius abc` | Defines the RADIUS server group configuration and enters RADIUS server group configuration mode. |
| **Step 4** | **server name** *name*<br>**Example:**<br>`Device(config-sg-radius)# server name serv1` | Associates the RADIUS server to the server group. |
| **Step 5** | {**ip** \| **ipv6**} **radius source-interface** *type number*<br>**Example:**<br>`Device(config-sg-radius)# ipv6 radius source-interface ethernet 0/0` | Specifies an interface to use for the source address in RADIUS server. |
| **Step 6** | **dscp** {**acct** *dscp_acct_value* \| **auth** *dscp_auth_value*}<br>**Example:**<br>`Device(config-sg-radius)# dscp auth 10 acct 20` | Configures DSCP marking for authentication and accounting on the radius server group.<br>• **acct** configures radius DSCP marking value for accounting. The valid range is from 1 to 63. The default value is 0.<br>• **auth** configures radius DSCP marking value for authentication. The valid range is from 1 to 63. The default value is 0. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-radius-server)# end` | Exits RADIUS server mode and enters privileged EXEC mode. |

# Configuring CoA on the Device

Follow these steps to configure CoA on a device. This procedure is required.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa server radius dynamic-author**<br>**Example:**<br>`Device(config)# aaa server radius dynamic-author` | Configures the device as an authentication, authorization, and accounting (AAA) server to facilitate interaction with an external policy server, and enters dynamic authorization local server configuration mode. |
| **Step 5** | **client** {*ip-address* \| *name*} [**vrf** *vrfname*] [**server-key** *string*]<br>**Example:**<br>`Device(config-locsvr-da-radius)# client client1 vrf vrf1` | Specifies a RADIUS client from which a device will accept CoA and disconnect requests. |
| **Step 6** | **server-key** [**0** \| **7**] *string*<br>**Example:**<br>`Device(config-locsvr-da-radius)# server-key your_server_key` | Configures the RADIUS key to be shared between a device and RADIUS clients. |
| **Step 7** | **port** *port-number*<br>**Example:**<br>`Device(config-locsvr-da-radius)# port 25` | Specifies the port on which a device listens for RADIUS requests from configured RADIUS clients. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **auth-type** {**any** \| **all** \| **session-key**}<br>**Example:**<br>`Device(config-locsvr-da-radius)# auth-type any` | Specifies the type of authorization the device uses for RADIUS clients.<br>The client must match all the configured attributes for authorization. |
| **Step 9** | **ignore server-key**<br>**Example:**<br>`Device(config-locsvr-da-radius)# ignore server-key` | (Optional) Configures the device to ignore the server-key. |
| **Step 10** | **exit**<br>**Example:**<br>`Device(config-locsvr-da-radius)# exit` | Exits dynamic authorization local server configuration mode and returns to global configuration mode. |
| **Step 11** | **authentication command bounce-port ignore**<br>**Example:**<br>`Device(config)# authentication command bounce-port ignore` | (Optional) Configures the device to ignore a CoA request to temporarily disable the port hosting a session. The purpose of temporarily disabling the port is to trigger a DHCP renegotiation from the host when a VLAN change occurs and there is no supplicant on the endpoint to detect the change. |
| **Step 12** | **authentication command disable-port ignore**<br>**Example:**<br>`Device(config)# authentication command disable-port ignore` | (Optional) Configures the device to ignore a nonstandard command requesting that the port hosting a session be administratively shut down. Shutting down the port results in termination of the session.<br>Use standard CLI or SNMP commands to re-enable the port. |
| **Step 13** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode an returns to privileged EXEC mode. |

# Configuring RADIUS Source-Interface Under a RADIUS Server-Group

The RADIUS source-interface can be configured under a RADIUS server-group in either of the following methods:

- Configure a RADIUS source-interface under the RADIUS server-group using the **ip radius source-interface** *interface-name* command.

- Configure a VRF using the **vrf** *vrf-name* command under the RADIUS server-group, and then associate the configured VRF globally to a source-interface using the **ip radius source interface** *interface-name* **vrf** *vrf-name* command.

Priority will be given to the source-interface under the server-group configuration in case both methods are configured.

To configure RADIUS source-interface under a RADIUS server-group, perform the following:

### Before you begin

You must configure a VRF routing table and associate VRF to an interface

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | { **ip** \| **ipv6** }**radius source-interface** *interface-number* **vrf** *vrf-name*<br>**Example:**<br>`Device(config)# ip radius source-interface GigabitEthernet1/0/23 vrf vrf17` | Forces RADIUS to use the IP address of a specified interface for all outgoing RADIUS packets, and enables the specification on a per-VRF basis.<br>• *interface-name*: Specifies the name of the interface that RADIUS uses for all of its outgoing packets.<br>• **vrf** *vrf-name*: Specifies the per-VRF configuration. |
| **Step 4** | **aaa group server radius** *group_name*<br>**Example:**<br>`Device(config-sg-radius)# aa group server radius rad-grp` | Groups different RADIUS server hosts into distinct lists and distinct methods and enters server-group configuration mode. |
| **Step 5** | **ip vrf forwarding** *vrf-name*<br>**Example:**<br>`Device(config-sg-radius)# ip vrf forwarding vrf17` | (Optional) Configures a VRF for the interface. |
| **Step 6** | { **ip** \| **ipv6** } **radius source-interface** *interface-number*<br>**Example:** | (Optional) Forces RADIUS to use the IP address of a specified interface for all outgoing RADIUS packets from the RADIUS group server. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-sg-radius)# ip radius source-interface loopback0` | *interface-name*: Specifies the name of the interface that RADIUS uses for all of its outgoing packets. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-sg-radius)# end` | Returns to privileged EXEC mode. |

# Monitoring CoA Functionality

*Table 16: Privileged EXEC show Commands*

| Command | Purpose |
|---|---|
| **show aaa attributes protocol radius** | Displays AAA attributes of RADIUS commands. |

*Table 17: Global Troubleshooting Commands*

| Command | Purpose |
|---|---|
| **debug radius** | Displays information for troubleshooting RADIUS. |
| **debug aaa coa** | Displays information for troubleshooting CoA processing. |
| **debug aaa pod** | Displays information for troubleshooting POD packets. |
| **debug aaa subsys** | Displays information for troubleshooting POD packets. |
| **debug cmdhd** [**detail** \| **error** \| **events**] | Displays information for troubleshooting command headers. |

# Feature History for RADIUS

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | RADIUS | RADIUS is a distributed client/server system that secures networks against unauthorized access. RADIUS clients run on supported Cisco devices. Clients send authentication requests to a central RADIUS server, which contains all user authentication and network service access information.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | RADIUS | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Bengaluru 17.5.1 | DSCP marking on Radius servers | This feature allows you to configure Differentiated Services Code Point (DSCP) marking on RADIUS servers and RADIUS server groups using **dscp** command. The **radius-server dscp** command is used to configure DSCP marking for authentication and accounting on RADIUS servers in global configuration mode. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 11

# Configuring RadSec

This chapter describes how to configure RadSec over Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) servers.


- Restrictions for Configuring RadSec, on page 191
- Information About RadSec, on page 192
- How to Configure RadSec, on page 192
- Monitoring RadSec, on page 197
- Configuration Examples for RadSec, on page 197
- Feature History for Configuring RadSec, on page 199


## Restrictions for Configuring RadSec

The following restrictions apply to the RadSec feature:

- A RADIUS client uses an ephemeral port as the source port. This source port should not be used for UDP, Datagram Transport Layer Security (DTLS), and Transport Layer Security (TLS) at the same time.
- Although there is no configuration restriction, we recommend that you use the same type, either only TLS or only DTLS, for a server under an AAA server group.
- RadSec is not supported on the DTLS port range 1 to 1024.

  DTLS ports must be configured to work with the Radius server.
- RadSec is not supported with high availability.
- RADIUS Change of Authorization (CoA) reception of request and transmission of response over the same authentication channel is supported with RadSec over TLS only. It is not supported over DTLS or plain RADIUS.
- The **tls watchdoginterval** functionality is not applicable for Packet of Disconnect (POD) use cases.
- FQDN configuration for CoA is not supported.

# Information About RadSec

RadSec provides encryption services over the RADIUS server transported over a secure tunnel. RadSec over TLS and DTLS is implemented in both client and device servers. While the client side controls RADIUS AAA, the device side controls CoA.

You can configure the following parameters:

- Individual client-specific idle timeout, client trustpoint, and server trustpoint.
- Global CoA-specific TLS or DTLS listening port and the corresponding list of source interfaces.

**Note** You can disable TLS or DTLS for a specific server by using the **no tls** or **no dtls** command in radius server configuration mode.

RadSec CoA request reception and CoA response transmission over the same authentication channel can be enabled by configuring the **tls watchdoginterval** command. The TLS watchdog timer must be lesser than the TLS idle timer so that the established tunnel remains active if RADIUS test authentication packets are seen before the idle timer expires. If the tunnel is torn down and **tls watchdoginterval** command is enabled, the tunnel gets re-established immediately. If **tls watchdoginterval** command is disabled, CoA requests on the same authentication channel are discarded.

# How to Configure RadSec

The following sections provide information about the various tasks that comprise RadSec configuration:

## Configuring RadSec over TLS

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **radius server** *radius-server-name*<br>**Example:**<br>`Device(config)# radius server R1` | Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **tls** [**connectiontimeout** *connection-timeout-value*] [**idletimeout** *idle-timeout-value*] [[**ip** \| **ipv6**] {**radius source-interface** *interface-name* \|**vrf forwarding** *forwarding-table-name*} ] [**match-server-identity** {**email-address** *email-address* \| **hostname** *host-name* \| **ip-address** *ip-address*}] [**port** *port-number*] [**retries** *number-of-connection-retries*] [**trustpoint** {**client** *trustpoint name* \| **server** *trustpoint name*}] [**watchdoginterval** *interval*]<br>**Example:**<br>`Device(config-radius-server)# tls connectiontimeout 10`<br>`Device(config-radius-server)# tls idletimeout 75`<br>`Device(config-radius-server)# tls retries 15`<br>`Device(config-radius-server)# tls ip radius source-interface GigabitEthernet 1/0/1`<br>`Device(config-radius-server)# tls ipv6 vrf forwarding table-1`<br>`Device(config-radius-server)# tls match-server-identity ip-address 10.1.1.10`<br>`Device(config-radius-server)# tls port 10`<br>`Device(config-radius-server)# tls trustpoint client TP-self-signed-721943660`<br>`Device(config-radius-server)# tls trustpoint server isetp`<br>`Device(config-radius-server)# tls watchdoginterval 10` | Configures the TLS parameters. You can configure the following parameters:<br>• **connectiontimeout**: Configures TLS connection timeout value. The default is 5 seconds.<br>• **idletimeout**: Configures the TLS idle timeout value. The default is 60 seconds.<br>• **ip**: Configures IP source parameters.<br>• **ipv6**: Configures IPv6 source parameters.<br>• **match-server-identity**: Configures RadSec certification validation parameters.<br>**Note**<br>This is a mandatory configuration.<br>• **port**: Configures the TLS port number. The default is 2083.<br>• **retries**: Configures the number of TLS connection retries. The default is 5.<br>• **trustpoint**: Configures the TLS trustpoint for a client and a server. If the TLS trustpoint for the client and server are the same, the trustpoint name should also be the same for both.<br>• **watchdoginterval**: Configures the watchdog interval. This enables CoA requests to be received on the same authentication channel. It also serves as keepalive to keep the TLS tunnel up, and re-establishes the tunnel if it is torn down.<br>**Note**<br>**watchdoginterval** value must be lesser than **idletimeout** for the established tunnel to remain up. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-radius-server)# end` | Exits RADIUS server configuration mode and returns to privileged EXEC mode. |

# Configuring Dynamic Authorization for TLS CoA

**Note** When the **tls watchdoginterval** command is enabled, the client IP configuration under **aaa server radius dynamic-author** will not be used. Instead, the key configured under **radius server** will be used for CoA transactions.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa server radius dynamic-author**<br>**Example:**<br>`Device(config)# aaa server radius dynamic-author` | Enters dynamic authorization local server configuration mode and specifies the RADIUS client from which a device accepts Change of Authorization (CoA) and disconnect requests. Configures the device as an AAA server to facilitate interaction with an external policy server. |
| **Step 4** | **client {ip-addr \| hostname}** [**tls** [**client-tp** *client-tp-name*] [ **idletimeout** *idletimeout-interval* ] [**server-key** *server-key*] [**server-tp** *server-tp-name*]]<br>**Example:**<br>`Device(config-locsvr-da-radius)# client 10.104.49.14 tls idletimeout 100 client-tp tls_ise server-tp tls_client server-key key1` | Configures the IP address or hostname of the AAA server client. You can configure the following optional parameters:<br>• **tls**: Enables TLS for the client.<br>  • **client-tp**: Configures the client trustpoint.<br>  • **idletimeout**: Configures the TLS idle timeout value.<br>  • **server-key**: Configures a RADIUS client server-key.<br>  • **server-tp**: Configures the server trustpoint. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-locsvr-da-radius)# end` | Exits dynamic authorization local server configuration mode and returns to privileged EXEC mode. |

# Configuring RadSec over DTLS

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **radius server** *radius-server-name*<br>**Example:**<br>`Device(config)# radius server R1` | Specifies the name for the RADIUS server configuration for Protected Access Credential (PAC) provisioning, and enters RADIUS server configuration mode. |
| **Step 4** | **dtls** [**connectiontimeout** *connection-timeout-value*] [**idletimeout** *idle-timeout-value*] [[**ip** \| **ipv6**] {**radius source-interface** *interface-name* \|**vrf forwarding** *forwarding-table-name*} ] [**match-server-identity** {**email-address** *email-address* \| **hostname** *host-name* \| **ip-address** *ip-address*}] [**port** *port-number*] [**retries** *number-of-connection-retries*] [**trustpoint** {**client** *trustpoint name* \| **server** *trustpoint name*}]<br>**Example:**<br>`Device(config-radius-server)# dtls connectiontimeout 10`<br>`Device(config-radius-server)# dtls idletimeout 75`<br>`Device(config-radius-server)# dtls retries 15`<br>`Device(config-radius-server)# dtls ip radius source-interface GigabitEthernet 1/0/1`<br>`Device(config-radius-server)# dtls ipv6 vrf forwarding table-1`<br>`Device(config-radius-server)# tls match-server-identity ip-address 10.1.1.10`<br>`Device(config-radius-server)# dtls port 10` | Configures DTLS parameters. You can configure the following parameters:<br>• **connectiontimeout**: Configures the DTLS connection timeout value. The default is 5 seconds.<br>• **idletimeout**: Configures the DTLS idle timeout value. The default is 60 seconds.<br>**Note**<br>When the idle timeout expires, and there are no transactions after the last idle timeout, then the DTLS session is closed. When the session is reestablished, restart the idle timer for it to work.<br>If the configured idle timeout is 30 seconds, when the timeout expires, the number of RADIUS DTLS transactions are checked. If the RADIUS DTLS packets are more than 0, the transaction counter is reset and the timer is started again.<br>• **ip**: Configures IP source parameters.<br>• **ipv6**: Configures IPv6 source parameters.<br>• **match-server-identity**: Configures RadSec certification validation parameters.<br>**Note** |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-radius-server)# dtls trustpoint client TP-self-signed-721943660`<br>`Device(config-radius-server)# dtls trustpoint server isetp` | This is a mandatory configuration.<br>• **port**: Configures the DTLS port number. The default is 2083.<br>• **retries**: Configures the number of DTLS connection retries. The default is 5.<br>• **trustpoint**: Configures the DTLS trustpoint for the client and the server. If the DTLS trustpoint for the client and server are the same, the trustpoint name should also be the same for both. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-radius-server)# end` | Exits RADIUS server configuration mode and returns to privileged EXEC mode. |

# Configuring Dynamic Authorization for DTLS CoA

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa server radius dynamic-author**<br>**Example:**<br>`Device(config)# aaa server radius dynamic-author` | Enters dynamic authorization local server configuration mode and specifies a RADIUS client from which a device accepts Change of Authorization (CoA) and disconnect requests. Configures the device as an AAA server to facilitate interaction with an external policy server. |
| **Step 4** | **client {ip-addr \| hostname}** [**dtls** [**client-tp** *client-tp-name*] [ **idletimeout** *idletimeout-interval* ] [**server-key** *server-key*] [**server-tp** *server-tp-name*]]<br>**Example:**<br>`Device(config-locsvr-da-radius)# client 10.104.49.14 dtls idletimeout 100` | Configures the IP address or hostname of the AAA server client. You can configure the following optional parameters:<br>• **tls**: Enables TLS for the client.<br>  • **client-tp**: Configures the client trustpoint. |

| | Command or Action | Purpose |
|---|---|---|
| | `client-tp tls_ise server-tp tls_client server-key key1` | • **idletimeout**: Configures the TLS idle timeout value.<br>• **server-key**: Configures a RADIUS client server-key.<br>• **server-tp**: Configures the server trustpoint. |
| **Step 5** | **dtls** {{**ip** \| **ipv6**} **radius source-interface** *interface-name* \| **port** *radius-dtls-server-port-number*}<br>**Example:**<br>`Device(config-locsvr-da-radius)# dtls ip radius source-interface GigabitEthernet 1/0/24`<br>`Device(config-locsvr-da-radius)# dtls port 100` | Configures the RADIUS CoA server. You can configure the following parameters:<br>• {**ip** \| **ipv6**} **radius source-interface** *interface-name*: Specifies the interface for the source address in the RADIUS CoA server.<br>• **port** *radius-dtls-server-port-number*: Specifies the port on which the local DTLS RADIUS server listens. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-locsvr-da-radius)# end` | Exits dynamic authorization local server configuration mode and returns to privileged EXEC mode. |

# Monitoring RadSec

Use the following commands to monitor TLS and DTLS server statistics:

***Table 18: Monitoring TLS and DTLS Server Statistics Commands***

| Command | Purpose |
|---|---|
| **show aaa servers** | Displays information related to TLS and DTLS servers. |
| **clear aaa counters servers radius** {*server id* \| **all**} | Clears the RADIUS TLS-specific or DTLS-specific statistics. |
| **debug radius radsec** | Enables RADIUS RadSec debugs. |

# Configuration Examples for RadSec

The following examples help you understand the RadSec configuration better:

# Example: Configuring RadSec over TLS

The following example shows how to configure RadSec over TLS:

```
Device> enable
Device# configure terminal
Device(config)# radius server R1
Device(config-radius-server)# tls connectiontimeout 10
Device(config-radius-server)# tls idletimeout 75
Device(config-radius-server)# tls retries 15
Device(config-radius-server)# tls ip radius source-interface GigabitEthernet 1/0/1
Device(config-radius-server)# tls ip vrf forwarding table-1
Device(config-radius-server)# tls port 10
Device(config-radius-server)# tls trustpoint client TP-self-signed-721943660
Device(config-radius-server)# tls trustpoint server isetp
Device(config-radius-server)# tls watchdoginterval 10
Device(config-radius-server)# end
```

# Example: Configuring Dynamic Authorization for TLS CoA

The following example shows how to configure dynamic authorization for TLS CoA:

```
Device> enable
Device# configure terminal
Device(config)# aaa server radius dynamic-author
Device(config-locsvr-da-radius)# client 10.104.49.14 tls idletimeout 100
client-tp tls_ise server-tp tls_client
Device(config-locsvr-da-radius)# end
```

# Example: Configuring RadSec over DTLS

The following example shows how to configure RadSec over DTLS:

```
Device> enable
Device# configure terminal
Device(config)# radius server R1
Device(config-radius-server)# dtls connectiontimeout 10
Device(config-radius-server)# dtls idletimeout 75
Device(config-radius-server)# dtls retries 15
Device(config-radius-server)# dtls ip radius source-interface GigabitEthernet 1/0/1
Device(config-radius-server)# dtls ip vrf forwarding table-1
Device(config-radius-server)# dtls port 10
Device(config-radius-server)# dtls trustpoint client TP-self-signed-721943660
Device(config-radius-server)# dtls trustpoint server isetp
Device(config-radius-server)# end
```

# Example: Configuring Dynamic Authorization for DTLS CoA

The following example shows how to configure dynamic authorization for DTLS CoA:

```
Device> enable
Device# configure terminal
Device(config)# aaa server radius dynamic-author
Device(config-locsvr-da-radius)# client 10.104.49.14 dtls idletimeout 100
client-tp dtls_ise server-tp dtls_client
Device(config-locsvr-da-radius)# dtls ip radius source-interface GigabitEthernet 1/0/24
```

```
Device(config-locsvr-da-radius)# dtls port 100
Device(config-locsvr-da-radius)# end
```

# Feature History for Configuring RadSec

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Bengaluru 17.4.1 | Configuring RadSec | RadSec provides encryption services over the RADIUS server, which is transported over a secure tunnel. |
| Cisco IOS XE Bengaluru 17.6.1 | Radsec CoA over Same Tunnel | RadSec CoA request reception and CoA response transmission can be done over the same authentication channel. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 12

# RADIUS Server Load Balancing

The RADIUS Server Load Balancing feature distributes authentication, authorization, and accounting (AAA) authentication and accounting transactions across RADIUS servers in a server group. These servers can share the AAA transaction load and thereby respond faster to incoming requests.

This module describes the RADIUS Server Load Balancing feature.


- Prerequisites for RADIUS Server Load Balancing, on page 201
- Restrictions for RADIUS Server Load Balancing, on page 201
- Information About RADIUS Server Load Balancing, on page 202
- How to Configure RADIUS Server Load Balancing, on page 204
- Configuration Examples for RADIUS Server Load Balancing, on page 207
- Additional References for RADIUS Server Load Balancing, on page 213
- Feature History for RADIUS Server Load Balancing, on page 213


## Prerequisites for RADIUS Server Load Balancing

- Authentication, authorization, and accounting (AAA) must be configured on the RADIUS server.
- AAA RADIUS server groups must be configured.
- RADIUS must be configured for functions such as authentication, accounting, or static route download.

## Restrictions for RADIUS Server Load Balancing

- Incoming RADIUS requests, such as Packet of Disconnect (POD) requests, are not supported.
- Load balancing is not supported on proxy RADIUS servers and for private server groups.

# Information About RADIUS Server Load Balancing

## RADIUS Server Load Balancing Overview

Load balancing distributes batches of transactions to RADIUS servers within a server group. Load balancing assigns each batch of transactions to the server with the lowest number of outstanding transactions in its queue. The process of assigning a batch of transactions is as follows:

1. The first transaction is received for a new batch.
2. All server transaction queues are checked.
3. The server with the lowest number of outstanding transactions is identified.
4. The identified server is assigned the next batch of transactions.

The batch size is a user-configured parameter. Changes in the batch size may impact CPU load and network throughput. As batch size increases, CPU load decreases and network throughput increases. However, if a large batch size is used, all available server resources may not be fully utilized. As batch size decreases, CPU load increases and network throughput decreases.

**Note** There is no set number for large or small batch sizes. A batch with more than 50 transactions is considered large and a batch with fewer than 25 transactions is considered small.

**Note** If a server group contains ten or more servers, we recommend that you set a high batch size to reduce CPU load.

## Transaction Load Balancing Across RADIUS Server Groups

You can configure load balancing either per-named RADIUS server group or for the global RADIUS server group. The load balancing server group must be referred to as “radius” in the authentication, authorization, and accounting (AAA) method lists. All public servers that are part of the RADIUS server group are then load balanced.

You can configure authentication and accounting to use the same RADIUS server or different servers. In some cases, the same server can be used for preauthentication, authentication, or accounting transactions for a session. The preferred server, which is an internal setting and is set as the default, informs AAA to use the same server for the start and stop record for a session regardless of the server cost. When using the preferred server setting, ensure that the server that is used for the initial transaction (for example, authentication), the preferred server, is part of any other server group that is used for a subsequent transaction (for example, accounting).

The preferred server is not used if one of the following criteria is true:

- The **load-balance method least-outstanding ignore-preferred-server** command is used.
- The preferred server is dead.

- The preferred server is in quarantine.
- The want server flag has been set, overriding the preferred server setting.

The want server flag, an internal setting, is used when the same server must be used for all stages of a multistage transaction regardless of the server cost. If the want server is not available, the transaction fails.

You can use the **load-balance method least-outstanding ignore-preferred-server** command if you have either of the following configurations:

- Dedicated authentication server and a separate dedicated accounting server
- Network where you can track all call record statistics and call record details, including start and stop records and records that are stored on separate servers

If you have a configuration where authentication servers are a superset of accounting servers, the preferred server is not used.

# RADIUS Server Status and Automated Testing

The RADIUS Server Load Balancing feature considers the server status when assigning batches. Transaction batches are sent only to live servers. We recommend that you test the status of all RADIUS load-balanced servers, including low usage servers (for example, backup servers).

Transactions are not sent to a server that is marked dead. A server is marked dead until its timer expires, at which time it moves to quarantine state. A server is in quarantine until it is verified alive by the RADIUS automated tester functionality.

To determine if a server is alive and available to process transactions, the RADIUS automated tester sends a request periodically to the server for a test user ID. If the server returns an Access-Reject message, the server is alive; otherwise the server is either dead or quarantined.

A transaction sent to an unresponsive server is failed over to the next available server before the unresponsive server is marked dead. We recommend that you use the retry reorder mode for failed transactions.

When using the RADIUS automated tester, verify that the authentication, authorization, and accounting (AAA) servers are responding to the test packets that are sent by the network access server (NAS). If the servers are not configured correctly, packets may be dropped and the server erroneously marked dead.

**Caution** We recommend that you use a test user that is not defined on the RADIUS server for the RADIUS server automated testing to protect against security issues that may arise if the test user is not correctly configured.

**Note** Use the **test aaa group** command to check load-balancing transactions.

# VRF-Aware RADIUS Automated Testing

The RADIUS automated tester function works at a server-level configuration. There is no group associated with the function. A VRF is a group level configuration. All the information related to the VRF and the source-interface configurations is maintained in a group structure. If information regarding the VRF and the

source-interface configurations is available in the global source-interface, automated tester can access it. If the information is not available at the global source-interface or the default VRF, automated tester marks the server as a dead server.

Starting with Cisco IOS XE Bengaluru 17.4.1, you can configure automated tester to be VRF aware. You can use the **vrf** keyword with the **automate-tester** command to enable automate-tester for a non-default VRF.

**Note** For VRF aware automate-tester to work, you have to configure **global config ipv4/ipv6 source interface** *interface-name* **vrf** *vrf-name* command.

# How to Configure RADIUS Server Load Balancing

## Enabling Load Balancing for a Named RADIUS Server Group

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa group server radius** *group-name*<br>**Example:**<br>`Device(config)# aaa group server radius rad-sg` | Enters server group configuration mode. |
| **Step 4** | **server ip-address [auth-port port-number] [acct-port port-number]**<br>**Example:**<br>`Device(config-sg-radius)# server 192.0.2.238 auth-port 2095 acct-port 2096` | Configures the IP address of the RADIUS server for the group server. |
| **Step 5** | **load-balance method least-outstanding** [**batch-size** *number*] [**ignore-preferred-server**]<br>**Example:**<br>`Device(config-sg-radius)# load-balance method least-outstanding batch-size 30` | Enables the least-outstanding load balancing for a named server group. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **end**<br>**Example:**<br>`Device(config-sg-radius)# end` | Exits server group configuration mode and returns to privileged EXEC mode. |

# Troubleshooting RADIUS Server Load Balancing

After configuring the RADIUS Server Load Balancing feature, you can monitor the idle timer, dead timer, and load balancing server selection or verify the server status by using a manual test command.

## Procedure

**Step 1** Use the **debug aaa test** command to determine when an idle timer or dead timer has expired, when test packets are sent, the status of the server, or to verify the server state.

The idle timer is used to check the server status and is updated with or without any incoming requests. Monitoring the idle timer helps to determine if there are nonresponsive servers and to keep the RADIUS server status updated to efficiently utilize available resources. For instance, an updated idle timer would help ensure that incoming requests are sent to servers that are alive.

The dead timer is used either to determine that a server is dead or to update a dead server's status appropriately.

Monitoring server selection helps to determine how often the server selection changes. Server selection is effective in analyzing if there are any bottlenecks, a large number of queued requests, or if only specific servers are processing incoming requests.

The following sample output from the **debug aaa test** command shows when the idle timer expired:

**Example:**

```
Device# debug aaa test

Jul 16 00:07:01: AAA/SG/TEST: Server (192.0.2.245:1700,1701) quarantined.
Jul 16 00:07:01: AAA/SG/TEST: Sending test request(s) to server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Sending 1 Access-Requests, 1 Accounting-Requests in current
batch.
Jul 16 00:07:01: AAA/SG/TEST(Req#: 1): Sending test AAA Access-Request.
Jul 16 00:07:01: AAA/SG/TEST(Req#: 1): Sending test AAA Accounting-Request.
Jul 16 00:07:01: AAA/SG/TEST: Obtained Test response from server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Obtained Test response from server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Necessary responses received from server (192.0.2.245:1700,1701)
Jul 16 00:07:01: AAA/SG/TEST: Server (192.0.2.245:1700,1701) marked ALIVE. Idle timer set
for 60 sec(s).
Jul 16 00:07:01: AAA/SG/TEST: Server (192.0.2.245:1700,1701) removed from quarantine.
```

**Step 2** Use the **debug aaa sg-server selection** command to determine the server that is selected for load balancing.

The following sample output from the **debug aaa sg-server selection** command shows five access requests being sent to a server group with a batch size of three:

**Example:**

```
Device# debug aaa sg-server selection

Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
```

```
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [3] transactions remaining in batch. Reusing server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [2] transactions remaining in batch. Reusing server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [1] transactions remaining in batch. Reusing server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: No more transactions in batch. Obtaining a new server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining a new least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Server[0] load: 3
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Server[1] load: 0
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Server[2] load: 0
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Selected Server[1] with load 0
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [3] transactions remaining in batch.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: Obtaining least loaded server.
Jul 16 03:15:05: AAA/SG/SERVER_SELECT: [2] transactions remaining in batch. Reusing server.
```

**Step 3** Use the **test aaa group** command to manually verify the RADIUS load-balanced server status.

The following sample output shows the response from a load-balanced RADIUS server that is alive when the username “test” does not match a user profile. The server is verified alive when it issues an Access-Reject response to an authentication, authorization, and accounting (AAA) packet generated using the **test aaa group** command.

**Example:**

```
Device# test aaa group SG1 test lab new-code

00:06:07: RADIUS/ENCODE(00000000):Orig. component type = INVALID
00:06:07: RADIUS/ENCODE(00000000): dropping service type, "radius-server attribute 6
on-for-login-auth" is off
00:06:07: RADIUS(00000000): Config NAS IP: 192.0.2.4
00:06:07: RADIUS(00000000): sending
00:06:07: RADIUS/ENCODE: Best Local IP-Address 192.0.2.141 for Radius-Server 192.0.2.176
00:06:07: RADIUS(00000000): Send Access-Request to 192.0.2.176:1645 id 1645/1, len 50
00:06:07: RADIUS:  authenticator CA DB F4 9B 7B 66 C8 A9 - D1 99 4E 8E A4 46 99 B4
00:06:07: RADIUS:  User-Password       [2]   18  *
00:06:07: RADIUS:  User-Name           [1]   6   "test"
00:06:07: RADIUS:  NAS-IP-Address      [4]   6   192.0.2.141
00:06:07: RADIUS: Received from id 1645/1 192.0.2.176:1645, Access-Reject, len 44
00:06:07: RADIUS:  authenticator 2F 69 84 3E F0 4E F1 62 - AB B8 75 5B 38 82 49 C3
00:06:07: RADIUS:  Reply-Message       [18]  24
00:06:07: RADIUS:   41 75 74 68 65 6E 74 69 63 61 74 69 6F 6E 20 66  [Authentication f]
00:06:07: RADIUS:   61 69 6C 75 72 65                               [failure]
00:06:07: RADIUS(00000000): Received from id 1645/1
00:06:07: RADIUS/DECODE: Reply-Message fragments, 22, total 22 bytes
```

# Enabling VRF Aware RADIUS Automated Testing

To enable RADIUS automated testing for a non-default VRF, perform the following procedure:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device>enable` | Enables privileged EXEC mode. Enter your password, if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **radius server** *name*<br>**Example:**<br>`Device(config)# radius server myserver` | Specifies the name of the RADIUS server configuration and enters RADIUS server configuration mode. |
| **Step 4** | **address** {**ipv4** \| **ipv6**} {*ip-address* \| *host-name*} **auth-port** *port-numer* **acct-port** *port-number*<br>**Example:**<br>`Device(config-radius-server)# address ipv4 192.0.2.1 auth-port 1812 acct-port 1813` | Configures the IPv4 address for the RADIUS server accounting and authentication parameters. |
| **Step 5** | **automate-tester username** *user* [**ignore-auth-port**] [**ignore-acct-port**] [**idle-time** *minutes*] **vrf** *vrf-name* or **automate-tester username** *user* **probe-on vrf** *vrf-name*<br>**Example:**<br>`Device(config-radius-server)# automate-tester username user1 idle-time 2 vrf VRF1`<br>or<br>`Device(config-radius-server)# automate-tester username user1 probe-on vrf VRF1` | Enables RADIUS automated testing for a non-default VRF. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-radius-server)# end` | Exits RADIUS server configuration mode and returns to privileged EXEC mode. |

# Configuration Examples for RADIUS Server Load Balancing

## Example: Enabling Load Balancing for a Named RADIUS Server Group

The following examples show load balancing enabled for a named RADIUS server group. These examples are shown in three parts: the current configuration of the RADIUS command output, debug output, and authentication, authorization, and accounting (AAA) server status information.

The following sample output shows the relevant RADIUS configuration:

```
Device# show running-config
.
.
.
aaa group server radius server-group1
 server 192.0.2.238 auth-port 2095 acct-port 2096
 server 192.0.2.238 auth-port 2015 acct-port 2016
 load-balance method least-outstanding batch-size 5
!
aaa authentication ppp default group server-group1
aaa accounting network default start-stop group server-group1
.
.
.
Device(config-sg-radius)# load-balance method least-outstanding batch-size 30
```

The lines in the current configuration of the preceding RADIUS command output are defined as follows:

- The **aaa group server radius** command shows the configuration of a server group with two member servers.
- The **load-balance** command enables load balancing for global RADIUS server groups with the batch size specified.
- The **aaa authentication ppp** command authenticates all PPP users using RADIUS.
- The **aaa accounting** command enables sending of all accounting requests to the AAA server when the client is authenticated and then disconnected using the **start-stop** keyword.

The show debug sample output below shows the selection of the preferred server and the processing of requests for the preceding configuration:

```
Device# show debug

*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002C):No preferred server available.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:No more transactions in batch. Obtaining a new
server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining a new least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Server[0] load:0
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Server[1] load:0
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Selected Server[0] with load 0
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:[5] transactions remaining in batch.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002C):Server (192.0.2.238:2095,2096) now being
 used as preferred server
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002D):No preferred server available.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:[4] transactions remaining in batch. Reusing
server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002D):Server (192.0.2.238:2095,2096) now being
 used as preferred server
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002E):No preferred server available.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:[3] transactions remaining in batch. Reusing
server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002E):Server (192.0.2.238:2095,2096) now being
 used as preferred server
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002F):No preferred server available.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:[2] transactions remaining in batch. Reusing
server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(0000002F):Server (192.0.2.238:2095,2096) now being
 used as preferred server
```

```
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(00000030):No preferred server available.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT:[1] transactions remaining in batch. Reusing
server.
*Feb 28 13:51:16.019:AAA/SG/SERVER_SELECT(00000030):Server (192.0.2.238:2095,2096) now being
 used as preferred server
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT(00000031):No preferred server available.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:No more transactions in batch. Obtaining a new
server.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Obtaining a new least loaded server.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Server[1] load:0
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Server[0] load:5
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Selected Server[1] with load 0
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:[5] transactions remaining in batch.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT(00000031):Server (192.0.2.238:2015,2016) now being
 used as preferred server
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT(00000032):No preferred server available.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:Obtaining least loaded server.
*Feb 28 13:51:16.023:AAA/SG/SERVER_SELECT:[4] transactions remaining in batch. Reusing
server.
.
.
.
```

The following sample output from the **show aaa servers** command shows the AAA server status for the named RADIUS server group configuration:

The sample output shows the status of two RADIUS servers. Both servers are alive, and no requests have been processed since the counters were cleared 0 minutes ago.

```
Device# show aaa servers

RADIUS: id 3, priority 1, host 9:76:239::219, auth-port 1812, acct-port 1813, hostname r6
State: current UP, duration 223000s, previous duration 301s
Dead: total time 682s, count 2
Platform State from SMD: current UP, duration 222972s, previous duration 258s
SMD Platform Dead: total time 702s, count 3
Platform State from WNCD (1) : current UP
Platform State from WNCD (2) : current UP
Platform State from WNCD (3) : current UP
Platform State from WNCD (4) : current UP
Platform State from WNCD (5) : current UP
Platform State from WNCD (6) : current UP
Platform State from WNCD (7) : current UP
Platform State from WNCD (8) : current UP, duration 2451264s, previous duration 258s
Platform Dead: total time 703s, count 3
Quarantined: No
Authen: request 68, timeouts 68, failover 0, retransmission 53


Sates defination:
State: current UP. ===> this is IOSD state
Platform State from SMD: current UP. ====> This is wired BINOS i,e SMD
Platform State from WNCD (1) : current UP ===> This is wireless BINOS i.e WNCD instance 1
Platform State from WNCD (2) : current UP. ===> This is wireless BINOS i.e WNCD instance 2
Platform State from WNCD (3) : current UP
Platform State from WNCD (4) : current UP
Platform State from WNCD (5) : current UP
Platform State from WNCD (6) : current UP
Platform State from WNCD (7) : current UP
Platform State from WNCD (8) : current UP. ===> This is wireless BINOS i.e WNCD instance 8
```

# Example: Monitoring Idle Timer

The following example shows idle timer and related server state for load balancing enabled for a named RADIUS server group. The current configuration of the RADIUS command output and debug command output are also displayed.

The following sample output shows the relevant RADIUS configuration:

```
Device(config)# do show run aaa

aaa group server radius server-group1
radius server server1
address ipv4 192.0.2.1 auth-port 1812 acct-port 1813
automate-tester username user1 idle-time 2 vrf VRF1
radius-server load-balance method least-outstanding batch-size 5
```

The lines in the current configuration of the preceding RADIUS command output are defined as follows:

- The **aaa group server radius** command shows the configuration of a server group.
- The **radius server** and **address** command defines the RADIUS server name and IP address of the RADIUS server with authorization and accounting ports specified.
- The **radius-server load-balance** command enables load balancing for the RADIUS server with the batch size specified.

The **show debug** sample output below shows test requests being sent to servers. The response to the test request sent to the server is received, the server is removed from quarantine as appropriate, the server is marked alive, and then the idle timer is reset.

```
Device# show debug

*Feb 28 13:52:20.835:AAA/SG/TEST:Server (192.0.2.238:2015,2016) quarantined.
*Feb 28 13:52:20.835:AAA/SG/TEST:Sending test request(s) to server (192.0.2.238:2015,2016)
*Feb 28 13:52:20.835:AAA/SG/TEST:Sending 1 Access-Requests, 1 Accounting-Requests in current
 batch.
*Feb 28 13:52:20.835:AAA/SG/TEST(Req#:1):Sending test AAA Access-Request.
*Feb 28 13:52:20.835:AAA/SG/TEST(Req#:1):Sending test AAA Accounting-Request.
*Feb 28 13:52:21.087:AAA/SG/TEST:Obtained Test response from server (192.0.2.238:2015,2016)
*Feb 28 13:52:22.651:AAA/SG/TEST:Obtained Test response from server (192.0.2.238:2015,2016)
*Feb 28 13:52:22.651:AAA/SG/TEST:Necessary responses received from server
(192.0.2.238:2015,2016)
*Feb 28 13:52:22.651:AAA/SG/TEST:Server (192.0.2.238:2015,2016) marked ALIVE. Idle timer
set for 60 secs(s).
*Feb 28 13:52:22.651:AAA/SG/TEST:Server (192.0.2.238:2015,2016) removed from quarantine.
.
.
.
```

# Example: Configuring the Preferred Server with the Same Authentication and Authorization Server

The following example shows an authentication server group and an authorization server group that use the same servers 209.165.200.225 and 209.165.200.226. Both server groups have the preferred server flag enabled.

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius authentication-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
```

```
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# exit
Device(config)# aaa group server radius accounting-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# end
```

When a preferred server is selected for a session, all transactions for that session will continue to use the original preferred server. The servers 209.165.200.225 and 209.165.200.226 are load balanced based on sessions rather than transactions.

# Example: Configuring the Preferred Server with Different Authentication and Authorization Servers

The following example shows an authentication server group that uses servers 209.165.200.225 and 209.165.200.226 and an authorization server group that uses servers 209.165.201.1 and 209.165.201.2. Both server groups have the preferred server flag enabled.

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius authentication-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# exit
Device(config)# aaa group server radius accounting-group
Device(config-sg-radius)# server 209.165.201.1 key radkey3
Device(config-sg-radius)# server 209.165.201.2 key radkey4
Device(config-sg-radius)# end
```

The authentication server group and the accounting server group do not share any common servers. A preferred server is never found for accounting transactions; therefore, authentication and accounting servers are load-balanced based on transactions. Start and stop records are sent to the same server for a session.

# Example: Configuring the Preferred Server with Overlapping Authentication and Authorization Servers

The following example shows an authentication server group that uses servers 209.165.200.225, 209.165.200.226, and 209.165.201.1 and an accounting server group that uses servers 209.165.201.1 and 209.165.201.2. Both server groups have the preferred server flag enabled.

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius authentication-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# server 209.165.201.1 key radkey3
Device(config-sg-radius)# exit
Device(config)# aaa group server radius accounting-group
Device(config-sg-radius)# server 209.165.201.1 key radkey3
Device(config-sg-radius)# server 209.165.201.2 key radkey4
Device(config-sg-radius)# end
```

If all servers have equal transaction processing capability, one-third of all authentication transactions are directed toward the server 209.165.201.1. Therefore, one-third of all accounting transactions are also directed toward the server 209.165.201.1. The remaining two-third of accounting transactions are load balanced equally

between servers 209.165.201.1 and 209.165.201.2. The server 209.165.201.1 receives fewer authentication transactions because the server 209.165.201.1 has outstanding accounting transactions.

# Example: Configuring the Preferred Server with Authentication Servers As a Subset of Authorization Servers

The following example shows an authentication server group that uses servers 209.165.200.225 and 209.165.200.226 and an authorization server group that uses servers 209.165.200.225, 209.165.200.226, and 209.165.201.1. Both server groups have the preferred server flag enabled.

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius authentication-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# exit
Device(config)# aaa group server radius accounting-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# server 209.165.201.1 key radkey3
Device(config-sg-radius)# end
```

One-half of all authentication transactions are sent to the server 209.165.200.225 and the other half to the server 209.165.200.226. Servers 209.165.200.225 and 209.165.200.226 are preferred servers for authentication and accounting transaction. Therefore, there is an equal distribution of authentication and accounting transactions across servers 209.165.200.225 and 209.165.200.226. The server 209.165.201.1 is relatively unused.

# Example: Configuring the Preferred Server with Authentication Servers As a Superset of Authorization Servers

The following example shows an authentication server group that uses servers 209.165.200.225, 209.165.200.226, and 209.165.201.1 and an authorization server group that uses servers 209.165.200.225 and 209.165.200.226. Both server groups have the preferred server flag enabled.

```
Device> enable
Device# configure terminal
Device(config)# aaa group server radius authentication-groupp
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# server 209.165.201.1 key radkey3
Device(config-sg-radius)# exit
Device(config)# aaa group server radius accounting-group
Device(config-sg-radius)# server 209.165.200.225 key radkey1
Device(config-sg-radius)# server 209.165.200.226 key radkey2
Device(config-sg-radius)# end
```

Initially, one-third of authentication transactions are assigned to each server in the authorization server group. As accounting transactions are generated for more sessions, accounting transactions are sent to servers 209.165.200.225 and 209.165.200.226 because the preferred server flag is on. As servers 209.165.200.225 and 209.165.200.226 begin to process more transactions, authentication transactions will start to be sent to server 209.165.201.1. Transaction requests authenticated by server 209.165.201.1 do not have any preferred server setting and are split between servers 209.165.200.225 and 209.165.200.226, which negates the use of the preferred server flag. This configuration should be used cautiously.

## Example: Enabling VRF Aware RADIUS Automated Testing

The following examples show how to enable automated testing for a non-default VRF on the RADIUS server:

```
Device(config)# radius server myserver
Device(config-radius-server)# address ipv4 192.0.2.1 auth-port 1812 acct-port 1813
Device(config-radius-server)# automate-tester username user1 idle-time 2 vrf VRF1
Device(config-radius-server)# end
```

```
Device(config)# radius server myserver
Device(config-radius-server)# address ipv4 192.0.2.1 auth-port 1812 acct-port 1813
Device(config-radius-server)# automate-tester username user1 probe-on vrf VRF1
Device(config-radius-server)# end
```

# Additional References for RADIUS Server Load Balancing

### Related Documents

| Related Topic | Document Title |
|---|---|
| RADIUS | "Configuring RADIUS" module in the *Security Configuration Guide* |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/techsupport |

# Feature History for RADIUS Server Load Balancing

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | RADIUS Server Load Balancing | The RADIUS Server Load Balancing feature distributes authentication, authorization, and accounting (AAA) authentication and accounting transactions across servers in a server group. These servers can share the AAA transaction load and thereby respond faster to incoming requests.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | RADIUS Server Load Balancing | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Bengaluru 17.4.1 | VRF Aware RADIUS Automated Testing | You can configure RADIUS automated testing for a non-default VRF. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 13

# MACsec Encryption


- Prerequisites for MACsec Encryption, on page 215
- Restrictions for MACsec Encryption, on page 215
- Information About MACsec Encryption, on page 216
- How to Configure MACsec Encryption, on page 224
- Configuration Examples for MACsec Encryption, on page 247
- Additional References for MACsec Encryption, on page 268
- Feature History for MACsec Encryption, on page 269


## Prerequisites for MACsec Encryption

### Prerequisites for MACsec Encryption

- Enable the **ssci-based-on-sci** command while configuring MACsec encryption on the device to allow interoperability with non-Cisco and non-IOS XE devices.
- Ensure that 802.1x authentication and AAA are configured on your device.

### Prerequisites for Certificate-Based MACsec

- Ensure that you have a Certificate Authority (CA) server configured for your network.
- Generate a CA certificate.
- Ensure that you have configured Cisco Identity Services Engine (ISE) Release 2.0.
- Ensure that both the participating devices, the CA server, and Cisco Identity Services Engine (ISE) are synchronized using Network Time Protocol (NTP). If time is not synchronized on all your devices, certificates will not be validated.

## Restrictions for MACsec Encryption

- MACsec configuration is not supported on EtherChannel ports. Instead, MACsec configuration can be applied on the individual member ports of an EtherChannel. To remove MACsec configuration, you must first unbundle the member ports from the EtherChannel, and then remove it from the individual member ports.

- MACsec with MKA is supported only on point-to-point links.
- GCM-AES-256 and XPN cipher suites (GCM-AES-XPN-128 and GCM-AES-XPN-256) are supported only with Network Advantage license.
- The MACsec Cipher announcement is not supported for MACsec Extended Packet Numbering (XPN) Ciphers and switch-to-switch MACsec connections.
- MACsec XPN Cipher Suites are not supported in switch-to-host MACsec connections.
- MACsec XPN Cipher Suites do not provide confidentiality protection with a confidentiality offset, and these together are not supported in switch-to-switch MACsec connections.
- As per IEEE standards, the maximum value of replay window is $2^{30-1}$ for MACsec XPN Cipher Suites. Even if you configure a higher value than this, it will be restricted to $2^{30-1}$ only.
- Certificate-based MACsec is supported only if the access-session is configured as closed or in multiple-host mode. None of the other configuration modes are supported.
- If the **dot1q tag vlan native** command is configured globally, the dot1x reauthentication will fail on trunk ports.
- MACsec is not supported with Multicast VPN (mVPN).
- MACsec switch-to-host connections are not supported on Software-Defined Access deployments.
- **should-secure** access mode is supported on switch-to-switch ports only using PSK authentication.

# Information About MACsec Encryption

The following sections provide information about MACsec encryption.

## Recommendations for MACsec Encryption

This section list the recommendations for configuring MACsec encryption:

- Use the confidentiality (encryption) offset as 0 in switch-to-host connections.
- Use Bidirectional Forwarding and Detection (BFD) timer value as 750 milliseconds for 10Gbps ports and 1.25 seconds for any port with speed above 10Gbps.
- Execute the **shutdown** command, and then the **no shutdown** command on a port, after changing any MKA policy or MACsec configuration for active sessions, so that the changes are applied to active sessions.
- Use Extended Packet Numbering (XPN) Cipher Suite for port speeds of 40Gbps and above.
- Set the connectivity association key (CAK) rekey overlap timer to 30 seconds or more.
- Do not use Cisco TrustSec Security Association Protocol (SAP) MACsec encryption for port speeds above 10Gbps.
- Do not enable both Cisco TrustSec SAP and uplink MKA at the same time on any interface.
- We recommend that you use MACsec MKA encryption.

# MACsec Encryption Overview

MACsec is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. Catalyst switches support 802.1AE encryption with MACsec Key Agreement (MKA) on switch-to-host links for encryption between the switch and host device. The switch also supports MACsec encryption for switch-to-switch (inter-network device) security using both Cisco TrustSec Network Device Admission Control (NDAC), Security Association Protocol (SAP) and MKA-based key exchange protocol.

**Note** When switch-to-switch MACSec is enabled, all traffic is encrypted, except the EAP-over-LAN (EAPOL) packets.

Link layer security can include both packet authentication between switches and MACsec encryption between switches (encryption is optional). Link layer security is supported on SAP-based MACsec.

***Table 19: MACsec Support on Switch Ports***

| Connections | MACsec Support |
|---|---|
| Switch-to-Host | MACsec MKA Encryption |
| Switch-to-Switch | MACsec MKA encryption (recommended)<br>Cisco TrustSec SAP |

Cisco TrustSec and Cisco SAP are meant only for switch-to-switch links and are not supported on switch ports connected to end hosts, such as PCs or IP phones. MKA is supported on switch-to-host facing links as well as switch-to-switch links. Host-facing links typically use flexible authentication ordering for handling heterogeneous devices with or without IEEE 802.1x, and can optionally use MKA-based MACsec encryption. Cisco NDAC and SAP are mutually exclusive with Network Edge Access Topology (NEAT), which is used for compact switches to extend security outside the wiring closet.

# Media Access Control Security and MACsec Key Agreement

MACsec, defined in 802.1AE, provides MAC-layer encryption over wired networks by using out-of-band methods for encryption keying. The MACsec Key Agreement (MKA) Protocol provides the required session keys and manages the required encryption keys. MKA and MACsec are implemented after successful authentication using the certificate-based MACsec or Pre Shared Key (PSK) framework.

A switch using MACsec accepts either MACsec or non-MACsec frames, depending on the policy associated with the MKA peer. MACsec frames are encrypted and protected with an integrity check value (ICV). When the switch receives frames from the MKA peer, it decrypts them and calculates the correct ICV by using session keys provided by MKA. The switch compares that ICV to the ICV within the frame. If they are not identical, the frame is dropped. The switch also encrypts and adds an ICV to any frames sent over the secured port (the access point used to provide the secure MAC service to a MKA peer) using the current session key.

The MKA Protocol manages the encryption keys used by the underlying MACsec protocol. The basic requirements of MKA are defined in 802.1x-REV. The MKA Protocol extends 802.1x to allow peer discovery with confirmation of mutual authentication and sharing of MACsec secret keys to protect data exchanged by the peers.

**Note** Starting with Cisco IOS XE 16.12.1 release, support for MKA with high availability has been introduced for Cisco Catalyst 9500 High Performance Series Switches. The high availability feature enables a pair of route processors to act as backup for each other. With high availability support for MKA if there is an active RP failure, the stand-by RP takes over existing MKA sessions in a minimally-disruptive switchover.

The EAP framework implements MKA as a newly defined EAP-over-LAN (EAPOL) packet. EAP authentication produces a master session key (MSK) shared by both partners in the data exchange. Entering the EAP session ID generates a secure connectivity association key name (CKN). The switch acts as the authenticator for both switch-to-switch and switch-to-host; and acts as the key server for switch-to-host. It generates a random secure association key (SAK), which is sent to the client partner. The client is never a key server and can only interact with a single MKA entity, the key server. After key derivation and generation, the switch sends periodic transports to the partner at a default interval of 2 seconds.

The packet body in an EAPOL Protocol Data Unit (PDU) is referred to as a MACsec Key Agreement PDU (MKPDU). MKA sessions and participants are deleted when the MKA lifetime (6 seconds) passes with no MKPDU received from a participant. For example, if a MKA peer disconnects, the participant on the switch continues to operate MKA until 6 seconds have elapsed after the last MKPDU is received from the MKA peer.

**Note** Integrity check value (ICV) indicator in MKPDU is optional. ICV is not optional when the traffic is encrypted.

## MKA Policies

You apply a defined MKA policy to an interface to enable MKA on the interface. Removing the MKA policy disables MKA on that interface. You can configure these options:

- Policy name, not to exceed 16 ASCII characters.
- Confidentiality (encryption) offset of 0, 30, or 50 bytes for each physical interface

## Definition of Policy-Map Actions

This section describes the policy-map actions and its definition:

- Activate: Applies a service template to the session.
- Authenticate: Starts authentication of the session.
- Authorize: Explicitly authorizes a session.
- Set-domain: Explicitly sets the domain of a client.
- Terminate: Terminates the method that is running, and deletes all the method details associated with the session.
- Deactivate: Removes the service-template applied to the session. If not applied, no action is taken.
- Set-timer: Starts a timer and gets associated with the session. When the timer expires, any action that needs to be started can be processed.
- Authentication-restart: Restarts authentication.

- Clear-session: Deletes a session.
- Pause: Pauses authentication.

Rest of the actions as self-explanatory and are associated with authentication.

## Virtual Ports

Use virtual ports for multiple secured connectivity associations on a single physical port. Each connectivity association (pair) represents a virtual port. In switch-to-switch, you can have only one virtual port per physical port. In switch-to-host, you can have a maximum of two virtual ports per physical port, of which one virtual port can be part of a data VLAN; the other must externally tag its packets for the voice VLAN. You cannot simultaneously host secured and unsecured sessions in the same VLAN on the same port. Because of this limitation, 802.1x multiple authentication mode is not supported.

The exception to this limitation is in multiple-host mode when the first MACsec supplicant is successfully authenticated and connected to a hub that is connected to the switch. A non-MACsec host connected to the hub can send traffic without authentication because it is in multiple-host mode. We do not recommend using multi-host mode because after the first successful client, authentication is not required for other clients.

Virtual ports represent an arbitrary identifier for a connectivity association and have no meaning outside the MKA Protocol. A virtual port corresponds to a separate logical port ID. Valid port IDs for a virtual port are 0x0002 to 0xFFFF. Each virtual port receives a unique secure channel identifier (SCI) based on the MAC address of the physical interface concatenated with a 16-bit port ID.

## MKA Statistics

Some MKA counters are aggregated globally, while others are updated both globally and per session. You can also obtain information about the status of MKA sessions. See Example: Displaying MKA Information, on page 262 for further information.

## Key Lifetime and Hitless Key Rollover

A MACsec key chain can have multiple pre-shared keys (PSK) each configured with a key id and an optional lifetime. A key lifetime specifies at which time the key expires. In the absence of a lifetime configuration, the default lifetime is unlimited. When a lifetime is configured, MKA rolls over to the next configured pre-shared key in the key chain after the lifetime is expired. Time zone of the key can be local or UTC. Default time zone is UTC.

You can Key rolls over to the next key within the same key chain by configuring a second key in the key chain and configuring a lifetime for the first key. When the lifetime of the first key expires, it automatically rolls over to the next key in the list. If the same key is configured on both sides of the link at the same time, then the key rollover is hitless, that is, key rolls over without traffic interruption.

On all participating devices, the MACsec key chain must be synchronised by using Network Time Protocol (NTP) and the same time zone must be used. If all the participating devices are not synchronized, the connectivity association key (CAK) rekey will not be initiated on all the devices at the same time.

**Note** The lifetime of the keys need to be overlapped in order to achieve hitless key rollover.

## Replay Protection Window Size

Replay protection is a feature provided by MACsec to counter replay attacks. Each encrypted packet is assigned a unique sequence number and the sequence is verified at the remote end. Frames transmitted through a Metro Ethernet service provider network are highly susceptible to reordering due to prioritization and load balancing mechanisms used within the network.

A replay window is necessary to support the use of MACsec over provider networks that reorder frames. Frames within the window can be received out of order, but are not replay protected. The default window size is 0, which enforces strict reception ordering. The replay window size can be configured in the range of 0 to $2^{32}$- 1. In case of XPN cipher suite, maximum replay window size is $2^{30}$- 1, and if a higher window size is configured, the window size gets restricted to $2^{30}$- 1. If the cipher suite is changed to a non-XPN cipher suite, then there is no restriction and the configured window size is used.

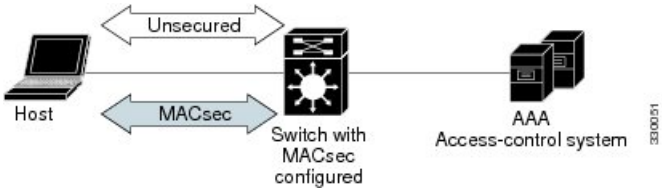
## MACsec, MKA, and 802.1x Host Modes

You can use MACsec and the MKA Protocol with 802.1x single-host mode, multi-host mode, or Multi Domain Authentication (MDA) mode. Multiple authentication mode is not supported.

### Single-Host Mode

The figure shows how a single EAP authenticated session is secured by MACsec by using MKA

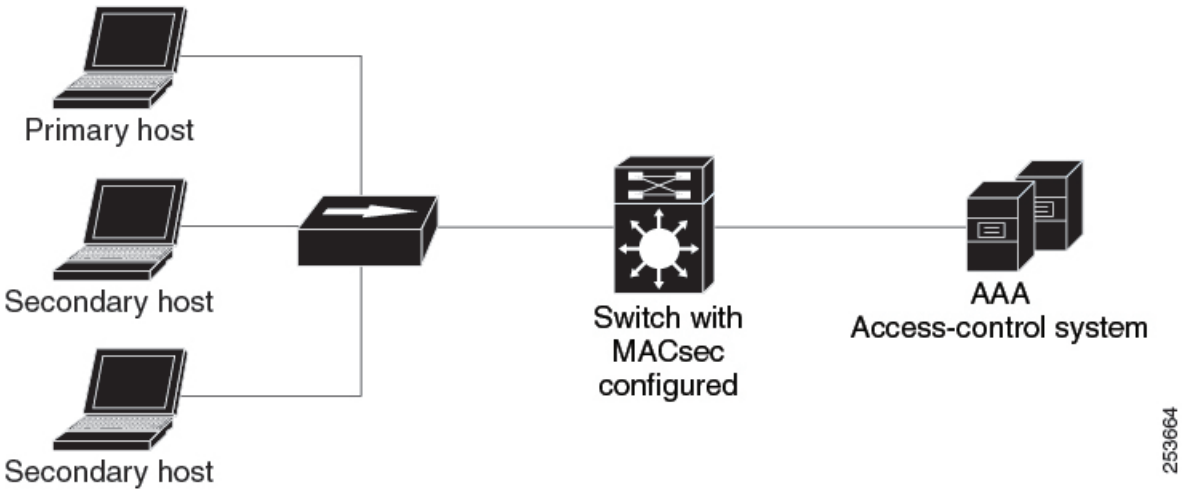
***Figure 12: MACsec in Single-Host Mode with a Secured Data Session***




### Multiple Host Mode

In standard (not 802.1x REV) 802.1x multiple-host mode, a port is open or closed based on a single authentication. If one user, the primary secured client services client host, is authenticated, the same level of network access is provided to any host connected to the same port. If a secondary host is a MACsec supplicant, it cannot be authenticated and traffic would not flow. A secondary host that is a non-MACsec host can send traffic to the network without authentication because it is in multiple-host mode. The figure shows MACsec in Standard Multiple-Host Unsecure Mode.

*Figure 13: MACsec in Multiple-Host Mode - Unsecured*




**Note** Multi-host mode is not recommended because after the first successful client, authentication is not required for other clients, which is not secure.

### Multiple-Domain Mode

In standard (not 802.1x REV) 802.1x multiple-domain mode, a port is open or closed based on a single authentication. If the primary user, a PC on data domain, is authenticated, the same level of network access is provided to any domain connected to the same port. If a secondary user is a MACsec supplicant, it cannot be authenticated and traffic would no flow. A secondary user, an IP phone on voice domain, that is a non-MACsec host, can send traffic to the network without authentication because it is in multiple-domain mode.

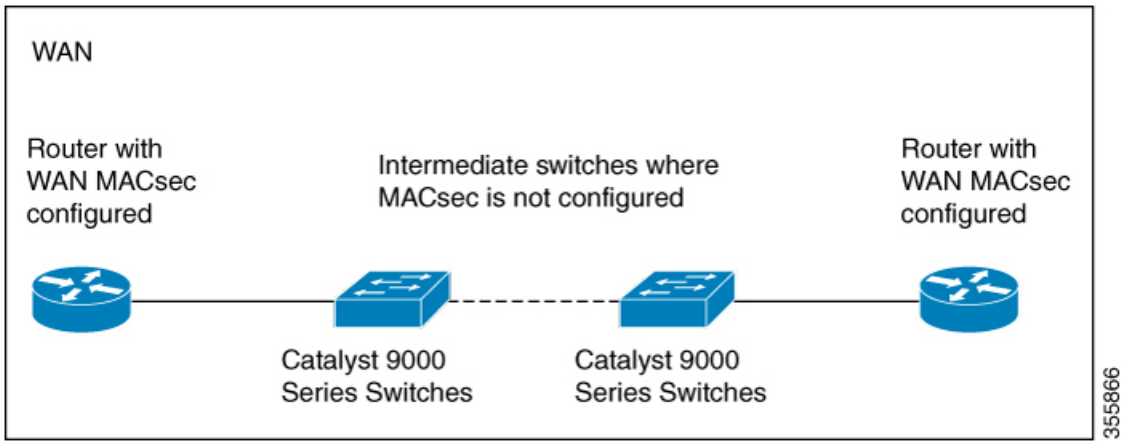
## Certificate-Based MACsec Encryption

Using certificate-based MACsec encryption, you can configure MACsec MKA between device switch-to-switch ports. Certificate-based MACsec encryption allows mutual authentication and obtains an MSK (master session key) from which the connectivity association key (CAK) is derived for MKA operations. Device certificates are carried, using certificate-based MACsec encryption, for authentication to the AAA server.

## MACsec Connections Across Intermediate Switches

Prior to Cisco IOS XE Gibraltar 16.10.1, MACsec connection between end devices which have WAN MACsec configured with the intermediate switches as the Cisco Catalyst 9000 Series Switches was not supported. The encrypted packets were dropped if WAN MACsec was configured on the end devices with MACsec not configured on the intermediate switches. With the ClearTag feature implemented on the ASIC, the switch forwards the encrypted packet without parsing the MACsec header. Below topology displays how the encrypted packets are forwarded through the intermediate switches with L2 switching.

*Figure 14: Topology for ClearTag MACsec : MACsec Not Configured on the Intermediate Switches*




## Limitations for MACsec Connections Across Intermediate Switches

- Hop-by-hop MACsec encryption with Catalyst 9000 Series switches as intermediate switches where WAN MACsec is configured on the routers is not supported.
- WAN MACsec configured on the routers with intermediate switches as the Catalyst 9000 Series switches is not supported on Layer 3 VPNs.
- WAN MACsec configured on the routers with intermediate switches as the Catalyst 9000 Series switches show Cisco Discovery Protocol neighbors only in should-secure mode.

# Switch-to-switch MKA MACsec Must Secure Policy

Starting with Cisco IOS XE Fuji 16.8.1a, must-secure support is enabled on both the ingress and the egress. Must-secure is supported for MKA and SAP. With must-secure enabled, only EAPoL traffic will not be encrypted. The rest of the traffic will be encrypted. Unencrypted packets are dropped.

**Note** Must-secure mode is enabled by default.

Prior to Cisco IOS XE Fuji 16.8.1a, should-secure was supported for MKA and SAP. With should-secure enabled, if the peer is configured for MACsec, the data traffic is encrypted, otherwise it is sent in clear text.

# MACsec Extended Packet Numbering (XPN)

Every MACsec frame contains a 32-bit packet number (PN), and it is unique for a given Security Association Key (SAK). Upon PN exhaustion (after reaching 75% of $2^{31}$- 1), SAK rekey takes place to refresh the data plane keys. For high capacity links such as 40 Gb/s, PN exhausts within a few seconds, and frequent SAK rekey to the control plane is required. When XPN is used, the PN of the MACsec frame is a 64-bit value, after reaching 75% of $^{th}$ of $2^{63}$- 1, it will require several years to exhaust the PN; this ensures that frequent SAK rekey does not happen on high speed links. The XPN feature in MKA/MACsec eliminates the need for frequent SAK rekey that may occur in high capacity links. XPN is a mandatory requirement for FIPS/CC compliance on high speed links such as 40 Gb/s, 100 Gb/s, and so on.

**Note** MACsec XPN is supported only on the switch-to-switch ports.

The following rekey is possible in XPN:

- **Volume-based Rekey**—To ensure that frequent SAK rekey does not happen, you can configure XPN using the GCM-AES-XPN-128 or GCM-AES-XPN-256 cipher suites under the defined MKA policy; these cipher suites allow more than $2^{32}$ frames to be protected with a single SAK. XPN supports a 64-bit value for the PN. The MACsec frame contains only the lowest 32 bits and the most significant 32 bits would be maintained by the peer itself, both the sending and the receiving peers. The most significant 32 bits of the PN is incremented at the receiving end when the MSB (most significant bits) of LAPN (lowest acceptable packet number) for the respective peer is set, and the MSB of the PN value received in the MACsec frame is 0. Thus, both the sending and the receiving peer maintain the same PN value without changing the MACsec frame structure.
- **Time-based Rekey**—To set the SAK rekey manually, timer-based rekey is supported where you have the provision to start re-keying SAK at a given interval. Use the **sak rekey interval** *time-interval* command in MKA policy configuration mode to configure the SAK rekey interval for a defined MKA policy applied to the interface.

# MKA/MACsec for Port Channel

MKA/MACsec can be configured on the port members of a port channel. MKA/MACsec is agnostic to the port channel since the MKA session is established between the port members of a port channel.

**Note** Etherchannel links that are formed as part of the port channel can either be congruent or disparate i.e. the links can either be MACsec-secured or non-MACsec-secured. MKA session between the port members is established even if a port member on one side of the port channel is not configured with MACsec.

It is recommended that you enable MKA/MACsec on all the member ports for better security of the port channel.

# MACsec Cipher Announcement

Cipher Announcement allows the supplicant and the authenticator to announce their respective MACsec Cipher Suite capabilities to each other. Both the supplicant and the authenticator calculate the largest common supported MACsec Cipher Suite and use the same as the keying material for the MKA session.

**Note** Only the MACsec Cipher Suite capabilities which are configured in the MKA policy are announced from the authenticator to the supplicant.

There are two types of EAPOL Announcements:

- Unsecured Announcements (EAPOL PDUs) : Unsecured announcements are EAPOL announcements carrying MACsec Cipher Suite capabilities in an unsecured manner. These announcements are used to decide the width of the key used for MKA session prior to authentication.

- Secure Announcements (MKPDUs) : Secure announcements revalidate the MACsec Cipher Suite capabilities which were shared previously through unsecure announcements.

Once the session is authenticated, peer capabilities which were received through EAPOL announcements are revalidated with the secure announcements. If there is a mismatch in the capabilities, the MKA session tears down.

**Note** The MKA session between the supplicant and the authenticator does not tear down even if the MACsec Cipher Suite Capabilities configured on both do not result in a common cipher suite.

# How to Configure MACsec Encryption

The following sections provide information about the various tasks that comprise MACsec encryption.

## Configuring MKA and MACsec

By default, MACsec is disabled. No MKA policies are configured.

### Configuring an MKA Policy

Beginning in privileged EXEC mode, follow these steps to create an MKA Protocol policy. Note that MKA also requires that you enable 802.1x.

#### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **mka policy** *policy-name*<br>**Example:**<br>`Device(config)# mka policy mka_policy` | Identifies an MKA policy, and enters MKA policy configuration mode. The maximum policy name length is 16 characters.<br>**Note**<br>The default MACsec cipher suite in the MKA policy will always be "GCM-AES-128". If the device supports both "GCM-AES-128" and "GCM-AES-256" ciphers, it is highly recommended to define and use a user defined MKA policy to include both 128 and 256 bits |

| | Command or Action | Purpose |
|---|---|---|
| | | ciphers or only 256 bits cipher, as may be required. |
| **Step 4** | **key-server** *priority*<br>**Example:**<br>`Device(config-mka-policy)# key-server priority 200` | Configures MKA key server options and set priority (between 0-255).<br>**Note**<br>When value of key server priority is set to 255, the peer can not become the key server. The key server priority value is valid only for MKA PSK; and not for MKA EAPTLS. |
| **Step 5** | **include-icv-indicator**<br>**Example:**<br>`Device(config-mka-policy)# include-icv-indicator` | Enables the ICV indicator in MKPDU. Use the **no** form of this command to disable the ICV indicator. |
| **Step 6** | **macsec-cipher-suite** {*gcm-aes-128* \| *gcm-aes-256*}<br>**Example:**<br>`Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128` | Configures a cipher suite for deriving SAK with 128-bit or 256-bit encryption. |
| **Step 7** | **confidentiality-offset** *offset-value*<br>**Example:**<br>`Device(config-mka-policy)# confidentiality-offset 0` | Set the confidentiality (encryption) offset for each physical interface.<br>**Note**<br>Offset Value can be 0, 30 or 50. If you are using Anyconnect on the client, it is recommended to use Offset 0. |
| **Step 8** | **ssci-based-on-sci**<br>**Example:**<br>`Device(config-mka-policy)# ssci-based-on-sci` | (Optional) Computes Short Secure Channel Identifier (SSCI) value based on Secure Channel Identifier (SCI) value. The higher the SCI value, the lower is the SSCI value. |
| **Step 9** | **end**<br>**Example:**<br>`Device(config-mka-policy)# end` | Exit enters MKA policy configuration mode and returns to privileged EXEC mode. |
| **Step 10** | **show mka policy**<br>**Example:**<br>`Device# show mka policy` | Displays MKA policy configuration information. |

## Configuring Switch-to-host MACsec Encryption

Follow these steps to configure MACsec on an interface with one MACsec session for voice and one for data:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter the password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters the global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface GigabitEthernet 1/0/1` | Identifies the MACsec interface, and enters interface configuration mode. The interface must be a physical interface. |
| **Step 4** | **switchport access vlan***vlan-id*<br>**Example:**<br>`Device(config-if)# switchport access vlan 1` | Configures the access VLAN for the port. |
| **Step 5** | **switchport mode access**<br>**Example:**<br>`Device(config-if)# switchport mode access` | Configures the interface as an access port. |
| **Step 6** | **macsec**<br>**Example:**<br>`Device(config-if)# macsec` | Enables 802.1ae MACsec on the interface. The macsec command enables MKA MACsec on switch-to-host links only. |
| **Step 7** | **authentication event linksec fail action authorize vlan** *vlan-id*<br>**Example:**<br>`Device(config-if)# authentication event linksec fail action authorize vlan 1` | (Optional) Specifies that the switch processes authentication link-security failures resulting from unrecognized user credentials by authorizing a restricted VLAN on the port after a failed authentication attempt. |
| **Step 8** | **authentication host-mode multi-domain**<br>**Example:**<br>`Device(config-if)# authentication host-mode multi-domain` | Configures authentication manager mode on the port to allow both a host and a voice device to be authenticated on the 802.1x-authorized port. If not configured, the default host mode is single. |
| **Step 9** | **authentication linksec policy must-secure**<br>**Example:**<br>`Device(config-if)# authentication linksec policy must-secure` | Sets the LinkSec security policy to secure the session with MACsec if the peer is available. If not set, the default is *should secure*. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 10** | **authentication port-control auto**<br>**Example:**<br>`Device(config-if)# authentication port-control auto` | Enables 802.1x authentication on the port. The port changes to the authorized or unauthorized state based on the authentication exchange between the switch and the client. |
| **Step 11** | **authentication periodic**<br>**Example:**<br>`Device(config-if)# authentication periodic` | (Optional) Enables or disables re-authentication for this port . |
| **Step 12** | **authentication timer reauthenticate**<br>**Example:**<br>`Device(config-if)# authentication timer reauthenticate` | (Optional) Enters a value between 1 and 65535 (in seconds). Obtains re-authentication timeout value from the server. Default re-authentication time is 3600 seconds. |
| **Step 13** | **authentication violation protect**<br>**Example:**<br>`Device(config-if)# authentication violation protect` | Configures the port to drop unexpected incoming MAC addresses when a new device connects to a port or when a device connects to a port after the maximum number of devices are connected to that port. If not configured, the default is to shut down the port. |
| **Step 14** | **mka policy** *policy-name*<br>**Example:**<br>`Device(config-if)# mka policy mka_policy` | Applies an existing MKA protocol policy to the interface, and enable MKA on the interface. If no MKA policy was configured (by entering the**mka policy** global configuration command). |
| **Step 15** | **dot1x pae authenticator**<br>**Example:**<br>`Device(config-if)# dot1x pae authenticator` | Configures the port as an 802.1x port access entity (PAE) authenticator. |
| **Step 16** | **spanning-tree portfast**<br>**Example:**<br>`Device(config-if)# spanning-tree portfast` | Enables spanning tree Port Fast on the interface in all its associated VLANs. When the Port Fast feature is enabled, the interface changes directly from a blocking state to a forwarding state without making the intermediate spanning-tree state changes |
| **Step 17** | **end**<br>**Example:**<br>`Device(config)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 18** | **show authentication session interface** *interface-id* **details**<br>**Example:**<br>`Device# show authentication session interface GigabitEthernet 1/0/1` | Verifies the details of the security status of the authorized session. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 19** | **show macsec interface** *interface-id*<br>**Example:**<br>`Device# show macsec interface GigabitEthernet 1/0/1` | Verifies the MACsec status on the interface. |
| **Step 20** | **show mka sessions**<br>**Example:**<br>`Device# show mka sessions` | Verifies the established MKA sessions. |

# Configuring MKA MACsec using PSK

## Configuring MACsec MKA using PSK

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **key chain** *key-chain-name* **macsec**<br>**Example:**<br>`Device(config)# key chain keychain1 macsec` | Configures a key chain and enters the key chain configuration mode. |
| **Step 4** | **key** *hex-string*<br>**Example:**<br>`Device(config-key-chain)# key 1000` | Configures a unique identifier for each key in the keychain and enters the keychain's key configuration mode.<br>**Note**<br>For 128-bit encryption, use any value between 1 and 32 hex digit key-string. For 256-bit encryption, use 64 hex digit key-string. |
| **Step 5** | **cryptographic-algorithm** *{aes-128-cmac \| aes-256-cmac}*<br>**Example:**<br>`Device(config-key-chain)# cryptographic-algorithm aes-128-cmac` | Set cryptographic authentication algorithm with 128-bit or 256-bit encryption. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **key-string** *{ [0\|6\|7] pwd-string \| pwd-string}*<br>**Example:**<br>`Device(config-key-chain)# key-string 12345678901234567890123456789012` | Sets the password for a key string. Only hex characters must be entered. |
| **Step 7** | **lifetime local** [*start timestamp {hh::mm::ss \| day \| month \| year}*] [**duration** *seconds \| end timestamp {hh::mm::ss \| day \| month \| year}*]<br>**Example:**<br>`Device(config-key-chain)# lifetime local 12:12:00 July 28 2016 12:19:00 July 28 2016` | Sets the lifetime of the pre shared key. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config-key-chain)# end` | Exits key chain configuration mode and returns to privileged EXEC mode. |

## Configuring MACsec MKA on an Interface using PSK

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config-if)# interface GigabitEthernet 0/0/0` | Enters interface configuration mode. |
| **Step 4** | **macsec network-link**<br>**Example:**<br>`Device(config-if)# macsec network-link` | Enables MACsec on the interface. |
| **Step 5** | **mka policy** *policy-name*<br>**Example:**<br>`Device(config-if)# mka policy mka_policy` | Configures an MKA policy. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **mka pre-shared-key key-chain** *key-chain name*<br>**Example:**<br>`Device(config-if)# mka pre-shared-key key-chain key-chain-name` | Configures an MKA pre-shared-key key-chain name. |
| **Step 7** | **macsec replay-protection window-size** *frame number*<br>**Example:**<br>`Device(config-if)# macsec replay-protection window-size 10` | Sets the MACsec window size for replay protection. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

### What to do next

It is not recommended to change the MKA policy on an interface with MKA PSK configured when the session is running. However, if a change is required, you must reconfigure the policy as follows:

1. Disable the existing session by removing macsec network-link configuration on each of the participating node using the **no macsec network-link** command
2. Configure the MKA policy on the interface on each of the participating node using the **mka policy policy-name** command.
3. Enable the new session on each of the participating node by using the **macsec network-link** command.

# Configuring Certificate-Based MACsec Encryption

To configure MACsec with MKA on point-to-point links, perform these tasks:

- Configure Certificate Enrollment
  - Generate Key Pairs
  - Configure SCEP Enrollment
  - Configure Certificates Manually
- Configure an Authentication Policy
- Configure certificate-based MACsec encryption Profiles and IEEE 802.1x Credentials
- Configure MKA MACsec using certificate-based MACsec encryption on Interfaces

## Generating Key Pairs

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **crypto key generate rsa label** *label-name* **general-keys modulus** *size*<br>**Example:**<br>`Device(config)# crypto key generate rsa label general-keys modulus 2048` | Generates a RSA key pair for signing and encryption.<br>You can also assign a label to each key pair using the label keyword. The label is referenced by the trustpoint that uses the key pair. If you do not assign a label, the key pair is automatically labeled <Default-RSA-Key>.<br>If you do not use additional keywords this command generates one general purpose RSA key pair. If the modulus is not specified, the default key modulus of 1024 is used. You can specify other modulus sizes with the modulus keyword. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |
| **Step 5** | **show authentication session interface** *interface-id*<br>**Example:**<br>`Device# show authentication session interface gigabitethernet 0/1/1` | Verifies the authorized session security status. |

## Configuring Enrollment using SCEP

Simple Certificate Enrollment Protocol (SCEP) is a Cisco-developed enrollment protocol that uses HTTP to communicate with the certificate authority (CA) or registration authority (RA). SCEP is the most commonly used method for sending and receiving requests and certificates.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **crypto pki trustpoint** *server name*<br>**Example:**<br>`Device(config)# crypto pki trustpoint ka` | Declares the trustpoint and a given name and enters ca-trustpoint configuration mode. |
| **Step 4** | **enrollment url** *url name pem*<br>**Example:**<br>`Device(ca-trustpoint)# enrollment url http://url:80` | Specifies the URL of the CA on which your device should send certificate requests.<br>An IPv6 address can be added in the URL enclosed in brackets. For example: http://[2001:DB8:1:1::1]:80.<br>The pem keyword adds privacy-enhanced mail (PEM) boundaries to the certificate request. |
| **Step 5** | **rsakeypair** *label*<br>**Example:**<br>`Device(ca-trustpoint)# rsakeypair exampleCAkeys` | Specifies which key pair to associate with the certificate.<br>**Note**<br>The **rsakeypair** name must match the trust-point name. |
| **Step 6** | **serial-number none**<br>**Example:**<br>`Device(ca-trustpoint)# serial-number none` | The **none** keyword specifies that a serial number will not be included in the certificate request. |
| **Step 7** | **ip-address none**<br>**Example:**<br>`Device(ca-trustpoint)# ip-address none` | The **none** keyword specifies that no IP address should be included in the certificate request. |
| **Step 8** | **revocation-check crl**<br>**Example:**<br>`Device(ca-trustpoint)# revocation-check crl` | Specifies CRL as the method to ensure that the certificate of a peer has not been revoked. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **auto-enroll** *percent* **regenerate**<br>**Example:**<br>`Device(ca-trustpoint)# auto-enroll 90 regenerate` | Enables auto-enrollment, allowing the client to automatically request a rollover certificate from the CA.<br>If auto-enrollment is not enabled, the client must be manually re-enrolled in your PKI upon certificate expiration.<br>By default, only the Domain Name System (DNS) name of the device is included in the certificate.<br>Use the percent argument to specify that a new certificate will be requested after the percentage of the lifetime of the current certificate is reached.<br>Use the regenerate keyword to generate a new key for the certificate even if a named key already exists.<br>If the key pair being rolled over is exportable, the new key pair will also be exportable. The following comment will appear in the trustpoint configuration to indicate whether the key pair is exportable: "! RSA key pair associated with trustpoint is exportable."<br>It is recommended that a new key pair be generated for security reasons. |
| **Step 10** | **exit**<br>**Example:**<br>`Device(ca-trustpoint)# exit` | Exits ca-trustpoint configuration mode and returns to global configuration mode. |
| **Step 11** | **crypto pki authenticate** *name*<br>**Example:**<br>`Device(config)# crypto pki authenticate myca` | Retrieves the CA certificate and authenticates it. |
| **Step 12** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |
| **Step 13** | **show crypto pki certificate** *trustpoint name*<br>**Example:**<br>`Device# show crypto pki certificate ka` | Displays information about the certificate for the trust point. |

## Configuring Enrollment Manually

If your CA does not support SCEP or if a network connection between the router and CA is not possible. Perform the following task to set up manual certificate enrollment:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **crypto pki trustpoint** *server name*<br>**Example:**<br>`Device# crypto pki trustpoint ka` | Declares the trustpoint and a given name and enters ca-trustpoint configuration mode. |
| **Step 4** | **enrollment url** *url-name*<br>**Example:**<br>`Device(ca-trustpoint)# enrollment url http://url:80` | Specifies the URL of the CA on which your device should send certificate requests.<br>An IPv6 address can be added in the URL enclosed in brackets. For example: http://[2001:DB8:1:1::1]:80.<br>The pem keyword adds privacy-enhanced mail (PEM) boundaries to the certificate request. |
| **Step 5** | **rsakeypair** *label*<br>**Example:**<br>`Device(ca-trustpoint)# rsakeypair exampleCAkeys` | Specifies which key pair to associate with the certificate. |
| **Step 6** | **serial-number none**<br>**Example:**<br>`Device(ca-trustpoint)# serial-number none` | Specifies that serial numbers will not be included in the certificate request. |
| **Step 7** | **ip-address none**<br>**Example:**<br>`Device(ca-trustpoint)# ip-address none` | The **none** keyword specifies that no IP address should be included in the certificate request. |
| **Step 8** | **revocation-check crl**<br>**Example:**<br>`Device(ca-trustpoint)# revocation-check crl` | Specifies CRL as the method to ensure that the certificate of a peer has not been revoked. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **exit**<br>**Example:**<br>`Device(ca-trustpoint)# exit` | Exits ca-trustpoint configuration mode and returns to global configuration mode. |
| **Step 10** | **crypto pki authenticate** *name*<br>**Example:**<br>`Device(config)# crypto pki authenticate myca` | Retrieves the CA certificate and authenticates it. |
| **Step 11** | **crypto pki enroll** *name*<br>**Example:**<br>`Device(config)# crypto pki enroll myca` | Generates certificate request and displays the request for copying and pasting into the certificate server.<br>Enter enrollment information when you are prompted. For example, specify whether to include the device FQDN and IP address in the certificate request.<br>You are also given the choice about displaying the certificate request to the console terminal.<br>The base-64 encoded certificate with or without PEM headers as requested is displayed. |
| **Step 12** | **crypto pki import** *name* **certificate**<br>**Example:**<br>`Device(config)# crypto pki import myca certificate` | Imports a certificate via TFTP at the console terminal, which retrieves the granted certificate.<br>The device attempts to retrieve the granted certificate via TFTP using the same filename used to send the request, except the extension is changed from ".req" to ".crt". For usage key certificates, the extensions "-sign.crt" and "-encr.crt" are used.<br>The device parses the received files, verifies the certificates, and inserts the certificates into the internal certificate database on the switch.<br>**Note**<br>Some CAs ignore the usage key information in the certificate request and issue general purpose usage certificates. If your CA ignores the usage key information in the certificate request, only import the general purpose certificate. The router will not use one of the two key pairs generated. |
| **Step 13** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 14** | **show crypto pki certificate** *trustpoint name*<br>**Example:**<br>`Device# show crypto pki certificate ka` | Displays information about the certificate for the trust point. |

## Configuring Switch-to-switch MACsec Encryption

To apply MACsec MKA using certificate-based MACsec encryption to interfaces, perform the following task:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 0/2/1` | Identifies the MACsec interface, and enters interface configuration mode. The interface must be a physical interface. |
| **Step 4** | **macsec network-link**<br>**Example:**<br>`Device(config-if)# macsec network-link` | Enables MACsec on the interface. |
| **Step 5** | **authentication periodic**<br>**Example:**<br>`Device(config-if)# authentication periodic` | Enables reauthentication for this port. |
| **Step 6** | **authentication timer reauthenticate interval**<br>**Example:**<br>`Device(config-if)# authentication timer reauthenticate interval` | Sets the reauthentication interval. |
| **Step 7** | **access-session host-mode multi-host**<br>**Example:**<br>`Device(config-if)# access-session host-mode multi-host` | Allows hosts to gain access to the interface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **access-session closed**<br>**Example:**<br>`Device(config-if)# access-session closed` | Prevents preauthentication access on the interface. |
| **Step 9** | **access-session port-control auto**<br>**Example:**<br>`Device(config-if)# access-session port-control auto` | Sets the authorization state of a port. |
| **Step 10** | **dot1x pae both**<br>**Example:**<br>`Device(config-if)# dot1x pae both` | Configures the port as an 802.1X port access entity (PAE) supplicant and authenticator. |
| **Step 11** | **dot1x credentials profile**<br>**Example:**<br>`Device(config-if)# dot1x credentials profile` | Assigns a 802.1x credentials profile to the interface. |
| **Step 12** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mdoe and returns to privileged EXEC mode. |
| **Step 13** | **show macsec interface** *interface-id*<br>**Example:**<br>`Device# show macsec interface GigabitEthernet 1/0/1` | Displays MACsec details for the interface. |

# Configuring MACsec XPN

## Configuring an MKA Policy for XPN

Follow these steps to configure XPN in an MKA policy:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **mka policy** *policy-name*<br>**Example:**<br>`Device(config)# mka policy mka_policy` | Identifies an MKA policy, and enters MKA policy configuration mode. The maximum policy name length is 16 characters.<br>**Note**<br>The default MACsec cipher suite in the MKA policy will always be "GCM-AES-128". If the device supports both "GCM-AES-128" and "GCM-AES-256" ciphers, it is highly recommended to define and use a user defined MKA policy to include both 128 and 256 bits ciphers or only 256 bits cipher, as may be required. |
| **Step 4** | **macsec-cipher-suite** { *gcm-aes-128* \| *gcm-aes-256* \| *gcm-aes-xpn-128* \| *gcm-aes-xpn-256*}<br>**Example:**<br>`Device(config-mka-policy)# macsec-cipher-suite gcm-aes-xpn-256` | Configures cipher suite for deriving SAK with 128-bit and 256-bit encryption for XPN. |
| **Step 5** | **sak-rekey interval** *time-interval*<br>**Example:**<br>`Device(config-mka-policy)# sak-rekey interval 50` | (Optional) Configures the SAK rekey interval (in seconds). The range is from 30 to 65535. By default, the SAK rekey interval occurs automatically depending on the interface speed.<br>Use the no form of this command to stop the SAK rekey timer. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-mka-policy)# end` | Exits MKA policy configuration mode and returns to privileged EXEC mode. |

## Applying the XPN MKA Policy to an Interface

To apply the XPN MKA policy to an interface, perform the following task:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **interface** *interface-name*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Identifies the MACsec interface, and enters interface configuration mode. The interface must be a physical interface. |
| **Step 4** | **mka policy** *policy-name*<br>**Example:**<br>`Device(config-if)# mka policy mka-xpn-policy` | Applies the XPN MKA protocol policy to the interface. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring MKA/MACsec for Port Channel

## Configuring MKA/MACsec for Port Channel using PSK

Beginning in privileged EXEC mode, follow these steps to configure MKA policies on an interface using a Pre-Shared Key (PSK).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config-if)# interface gigabitethernet 1/0/3` | Enters interface configuration mode. |
| **Step 4** | **macsec network-link**<br>**Example:**<br>`Device(config-if)# macsec network-link` | Enables MACsec on the interface. Supports layer 2 and layer 3 port channels. |
| **Step 5** | **mka policy** *policy-name*<br>**Example:**<br>`Device(config-if)# mka policy mka_policy` | Configures an MKA policy. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **mka pre-shared-key key-chain** *key-chain-name*<br>**Example:**<br>`Device(config-if)# mka pre-shared-key key-chain key-chain-name` | Configures an MKA pre-shared key key-chain name.<br>**Note**<br>The MKA pre-shared key can be configured either on the physical interface or the subinterface but not on both. |
| **Step 7** | **macsec replay-protection window-size** *frame number*<br>**Example:**<br>`Device(config-if)# macsec replay-protection window-size 0` | Sets the MACsec window size for replay protection. |
| **Step 8** | **channel-group** *channel-group-number* **mode** {**auto** \| **desirable**} \| {**active** \| **passive**} \| {**on**}<br>**Example:**<br>`Device(config-if)# channel-group 3 mode auto active on` | Configures the port in a channel group and sets the mode.<br>**Note**<br>You cannot configure ports in a channel group without configuring MACsec on the interface. You must configure the commands in Step 3, 4, 5 and 6 before this step.<br><br>The channel-number range is from 1 to 4096. The port channel associated with this channel group is automatically created if the port channel does not already exist. For mode, select one of the following keywords:<br>• **auto**— Enables PAgP only if a PAgP device is detected. This places the port into a passive negotiating state, in which the port responds to PAgP packets it receives but does not start PAgP packet negotiation.<br>**Note**<br>The **auto** keyword is not supported when EtherChannel members are from different switches in the switch stack.<br>• **desirable** — Unconditionally enables PAgP. This places the port into an active negotiating state, in which the port starts negotiations with other ports by sending PAgP packets.<br>**Note**<br>The **desirable** keyword is not supported when EtherChannel members are from different switches in the switch stack. |

| | Command or Action | Purpose |
|---|---|---|
| | | • **on**— Forces the port to channel without PAgP or LACP. In the **on** mode, an EtherChannel exists only when a port group in the **on** mode is connected to another port group in the **on** mode.<br>• **active**— Enables LACP only if an LACP device is detected. It places the port into an active negotiating state, in which the port starts negotiations with other ports by sending LACP packets.<br>• **passive**— Enables LACP on the port and places it into a passive negotiating state in which the port responds to LACP packets that it receives, but does not start LACP packet negotiation. |
| **Step 9** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

## Configuring Port Channel Logical Interfaces for Layer 2 EtherChannels

To create a port channel interface for a Layer 2 EtherChannel, perform this task:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface port-channel** *channel-group-number*<br>**Example:**<br>`Device(config)# interface port-channel 1` | Creates the port channel interface, and enters interface configuration mode.<br>**Note**<br>Use the **no** form of this command to delete the port channel interface. |
| **Step 4** | **switchport**<br>**Example:**<br>`Device(config-if)# switchport` | Switches an interface that is in Layer 3 mode into Layer 2 mode for Layer 2 configuration. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **switchport mode** {**access** \| **trunk**}<br>**Example:**<br>`Device(config-if)# switchport mode access` | Assigns all ports as static-access ports in the same VLAN, or configure them as trunks. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

## Configuring Port Channel Logical Interfaces for Layer 3 EtherChannels

To create a port channel interface for a Layer 3 EtherChannel, perform this task:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/2` | Enters interface configuration mode. |
| **Step 4** | **no switchport**<br>**Example:**<br>`Device(config-if)# no switchport` | Switches an interface that is in Layer 2 mode into Layer 3 mode for Layer 3 configuration. |
| **Step 5** | **ip address** *ip-address subnet-mask*<br>**Example:**<br>`Device(config-if)# ip address 10.2.2.3 255.255.255.254` | Assigns an IP address and subnet mask to the EtherChannel. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring MACsec Cipher Announcement

The following sections provide information about the various tasks to configure MACsec cipher announcement.

## Configuring an MKA Policy for Secure Announcement

Beginning in privileged EXEC mode, follow these steps to create an MKA Protocol policy to enable secure announcement in MKPDUs. By default, secure announcements are disabled.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **mka policy** *policy-name*<br>**Example:**<br>`Device(config)# mka policy mka_policy` | Identifies an MKA policy and enters MKA policy configuration mode. The maximum policy name length is 16 characters.<br>**Note**<br>The default MACsec cipher suite in the MKA policy is GCM-AES-128. If the device supports both GCM-AES-128 and GCM-AES-256 ciphers, we recommend that you define and use a user-defined MKA policy to include both 128 and 256 bits ciphers or only 256 bits cipher, as may be required. |
| **Step 4** | **key-server** *priority*<br>**Example:**<br>`Device(config-mka-policy)# key-server priority 200` | Configures MKA key server options and sets priority between 0-255.<br>**Note**<br>When value of key server priority is set to 255, the peer cannot become the key server. The key server priority value is valid only for MKA PSK. This does not apply to MKA EAP-TLS. |
| **Step 5** | **send-secure-announcements**<br>**Example:**<br>`Device(config-mka-policy)# send-secure-announcements` | Enables sending of secure announcements. Use the **no** form of the command to disable sending of secure announcements. By default, secure announcements are disabled. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **macsec-cipher-suite** {*gcm-aes-128* \| *gcm-aes-256*}<br>**Example:**<br>`Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128` | Configures cipher suite for deriving SAK with 128-bit or 256-bit encryption. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-mka-policy)# end` | Exits MKA policy configuration mode and returns to privileged EXEC mode. |
| **Step 8** | **show mka policy**<br>**Example:**<br>`Device# show mka policy` | Displays MKA policies. |

## Configuring Secure Announcement Globally

Beginning in privileged EXEC mode, follow these steps to enable secure announcement globally across all the MKA policies.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **mka defaults policy send-secure-announcements**<br>**Example:**<br>`Device(config)# mka defaults policy send-secure-announcements` | Enables sending of secure announcements in MKPDUs across MKA policies. By default, secure announcements are disabled. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

## Configuring EAPOL Announcements on an Interface

Beginning in privileged EXEC mode, follow these steps to configure EAPOL Announcement on an interface.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Identifies the MACsec interface, and enters interface configuration mode. The interface must be a physical interface. |
| **Step 4** | **eapol annoucement**<br>**Example:**<br>`Device(config-if)# eapol announcement` | Enables EAPOL announcements. Use the **no** form of the command to disable EAPOL announcements. By default, EAPOL announcements are disabled. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# configure terminal` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring Cisco TrustSec MACsec

## Configuring Cisco TrustSec Switch-to-Switch Link Security in Manual Mode

### Before you begin

When manually configuring Cisco TrustSec on an interface, consider these usage guidelines and restrictions:

- If no SAP parameters are defined, Cisco TrustSec encapsulation or encryption is not performed.
- If you select GCM as the SAP operating mode, you must have a MACsec Encryption software license from Cisco. If you select GCM without the required license, the interface is forced to a link-down state.
- These protection levels are supported when you configure SAP pairwise master key (sap pmk):
  - SAP is not configured: no protection.
  - **sap mode-list gcm-encrypt gmac no-encap**: protection desirable but not mandatory.
  - **sap mode-list gcm-encrypt gmac**: confidentiality preferred and integrity required. The protection is selected by the supplicant according to supplicant preference.
  - **sap mode-list gmac**: integrity only.

- **sap mode-list gcm-encrypt**: confidentiality required.
- **sap mode-list gmac gcm-encrypt**: integrity required and preferred, confidentiality optional.

- Before changing the configuration from MKA to Cisco TrustSec SAP and vice versa, we recommend that you remove the interface configuration.

Beginning in privileged EXEC mode, follow these steps to manually configure Cisco TrustSec on an interface to another Cisco TrustSec device:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 2** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface tengigabitethernet 1/1/2` | Configures an interface, and enters interface configuration mode. |
| **Step 3** | **cts manual**<br>**Example:**<br>`Device(config-if)# cts manual` | Enters Cisco TrustSec manual configuration mode. |
| **Step 4** | **sap pmk** *key* [**mode-list** *mode1* [*mode2* [*mode3* [*mode4*]]]]<br>**Example:**<br>`Device(config-if-cts-manual)# sap pmk 1234abcdef mode-list gcm-encrypt no-encap` | (Optional) Configures the SAP pairwise master key (PMK) and operation mode. SAP is disabled by default in Cisco TrustSec manual mode.<br>• *key*: A hexadecimal value with an even number of characters and a maximum length of 32 characters.<br>The SAP operation mode options:<br>• **gcm-encrypt**: Authentication and encryption<br>**Note**<br>Select this mode for MACsec authentication and encryption if your software license supports MACsec encryption.<br>• **gmac**: Authentication, no encryption<br>• **no-encap**: No encapsulation |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **no propagate sgt**<br>**Example:**<br>`Device(config-if-cts-manual)# no propagate sgt` | Use the **no** form of this command when the peer is incapable of processing a SGT. The **no propagate sgt** command prevents the interface from transmitting the SGT to the peer. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(config-if-cts-manual)# exit` | Exits Cisco TrustSec 802.1x interface configuration mode. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-if)# end` | Returns to privileged EXEC mode. |
| **Step 8** | **show cts interface** [*interface-id* \| **brief** \| **summary**] | (Optional) Verify the configuration by displaying TrustSec-related interface characteristics. |
| **Step 9** | **copy running-config startup-config**<br>**Example:**<br>`Device# copy running-config startup-config` | (Optional) Saves your entries in the configuration file. |

# Configuration Examples for MACsec Encryption

The following sections provide configuration examples for MACsec encryption.

## Example: Configuring MKA and MACsec

This example shows how to create an MKA policy:

```
Device> enable
Device# configure terminal
Device(config)# mka policy mka_policy
Device(config-mka-policy)# key-server priority 200
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 30
Device(config-mka-policy)# ssci-based-on-sci
Device(config-mka-policy)#end
```

This example shows how to configure MACsec on an interface:

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# switchport access vlan 1
Device(config-if)# switchport mode access
Device(config-if)# macsec
Device(config-if)# authentication event linksec fail action authorize vlan 1
Device(config-if)# authentication host-mode multi-domain
```

```
Device(config-if)# authentication linksec policy must-secure
Device(config-if)# authentication port-control auto
Device(config-if)# authentication periodic
Device(config-if)# authentication timer reauthenticate
Device(config-if)# authentication violation protect
Device(config-if)# mka policy mka_policy
Device(config-if)# dot1x pae authenticator
Device(config-if)# spanning-tree portfast
Device(config-if)# end
```

# Example: Configuring MACsec MKA using PSK

This example shows how to configure MACsec MKA using PSK.

```
Device> enable
Device# configure terminal
Device(config)# key chain keychain1 macsec
Device(config-keychain)# key 1000
Device(config-keychain-key)# cryptographic-algorithm aes-128-cmac
Device(config-keychain-key)# key-string 12345678901234567890123456789012
Device(config-keychain-key)# lifetime local 12:12:00 July 28 2016 12:19:00 July 28 2016
Device(config-keychain-key)# end
```

This example shows how to configure MACsec MKA on an interface using PSK.

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# mka policy mka_policy
Device(config-if)# mka pre-shared-key key-chain key-chain-name
Device(config-if)# macsec replay-protection window-size 10
Device(config-if)# end
```

### MKA-PSK: CKN Behavior Change

Starting Cisco IOS XE Fuji 16.8.1 release, for MKA PSK sessions, the CKN uses exactly the same string as the CKN which is configured as the hex-string for the key, instead of the fixed 32 bytes.

```
Device> enable
Device# configure terminal
Device(config)# key chain abc macsec
Device(config-keychain)# key 11
Device(config-keychain-key)# cryptographic-algorithm aes-128-cmac
Device(config-keychain-key)# key-string 12345678901234567890123456789013
Device(config-keychain-key)# lifetime local 12:21:00 Sep 9 2015 infinite
Device(config-keychain-key)# end
```

The following is sample output of the **show mka session** command for the above configuration:

```
Device# show mka session

Total MKA Sessions....... 1
Secured Sessions... 1
Pending Sessions... 0
=============================================================================================
Interface      Local-TxSCI         Policy-Name      Inherited         Key-Server

Port-ID        Peer-RxSCI          MACsec-Peers     Status            CKN

=============================================================================================
Et0/0          aabb.cc00.6600/0002     icv            NO                NO
```

```
2             aabb.cc00.6500/0002 1               Secured            11   *Note that the
 CKN key-string is exactly the same that has been configured for the key as hex-string.*
```

In case of interoperability between two images, where one having the CKN behavior change, and one without the CKN behavior change, the hex-string for the key must be a 64-character hex-string with zero padded for it to work on a device that has an image with the CKN behavior change. See the examples below:

Configuration without CKN key-string behavior change:

```
Device# configure terminal
Device(config)# key chain abc macsec
Device(config-keychain)# key 11
Device(config-keychain-key)# cryptographic-algorithm aes-128-cmac
Device(config-keychain-key)# key-string 12345678901234567890123456789013
Device(config-keychain-key)# lifetime local 12:21:00 Sep 9 2015 infinite
Device(config-keychain-key)# end
```

Configuration with CKN key-string behavior change:

```
Device# configure terminal
Device(config)# key chain abc macsec
Device(config-keychain)# key 1100000000000000000000000000000000000000000000000000000000000000
Device(config-keychain-key)# cryptographic-algorithm aes-128-cmac
Device(config-keychain-key)# key-string 12345678901234567890123456789013
Device(config-keychain-key)# lifetime local 12:21:00 Sep 9 2015 infinite
Device(config-keychain-key)# end
```

# Example: Configuring MACsec MKA Using Certificate-Based MACsec Encryption

This example shows how to configure MACsec MKA using certificate-based MACsec encryption:

```
Device> enable
Device# configure terminal
Device(config)# interface Gigabitethernet 1/0/1
Device(config-if)# macsec network-link
Device(config-if)# authentication periodic
Device(config-if)# authentication timer reauthenticate interval
Device(config-if)# access-session host-mode multi-domain
Device(config-if)# access-session closed
Device(config-if)# access-session port-control auto
Device(config-if)# dot1x pae both
Device(config-if)#dot1x credentials profile
Device(config-if)# dot1x supplicant eap profile profile_eap_tls
Device(config-if)# end
```

# Example: Configuring MACsec XPN

This example shows how to configure MACsec MKA XPN policy:

```
Device> enable
Device# configure terminal
Device(config)# mka policy mka-xpn-policy
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-xpn-256
Device(config-mka-policy)# end
```

This example shows how to apply MACsec MKA XPN policy to an interface:

```
Device> enable
Device# configure terminal
```

```
Device(config)#interface Fo 1/0/1
Device(config-if)# mka policy mka-xpn-policy
Device(config-if)# end
```

The following is a sample output of the **show mka sessions details** command with 128-bit XPN Cipher Suite configured:

```
Device# show mka sessions details

MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec

Local Tx-SCI............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)........... 0100000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89572
EAP Role................. NA
Key Server............... YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (1)
Old SAK Status........... FIRST-SAK
Old SAK AN............... 0
Old SAK KI (KN).......... FIRST-SAK (0)

SAK Transmit Wait Time... 0s (Not waiting for any peers to respond)
SAK Retire Time.......... 0s (No Old SAK to retire)

MKA Policy Name.......... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000003 (GCM-AES-XPN-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

# of MACsec Capable Live Peers............ 1
# of MACsec Capable Live Peers Responded.. 1

Live Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89560       c800.8459.e764/002a   10

Potential Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------

Dormant Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
```

The following is sample output of the **show mka sessions details** command with 256-bit XPN Cipher Suite configured:

```
Device# show mka sessions details

MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec

Local Tx-SCI............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)........... 0100000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89572
EAP Role................. NA
Key Server............... YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (1)
Old SAK Status........... FIRST-SAK
Old SAK AN............... 0
Old SAK KI (KN).......... FIRST-SAK (0)

SAK Transmit Wait Time... 0s (Not waiting for any peers to respond)
SAK Retire Time.......... 0s (No Old SAK to retire)

MKA Policy Name.......... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000004 (GCM-AES-XPN-256)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

# of MACsec Capable Live Peers............ 1
# of MACsec Capable Live Peers Responded.. 1

Live Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89560       c800.8459.e764/002a   10

Potential Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------

Dormant Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
```

# Example: Configuring MACsec MKA for Port Channel using PSK

### Etherchannel Mode — Static/On

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode on:

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
Device(config-key-chain)# key 1000
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# channel-group 2 mode on
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/0/2
Device(config-if)# channel-group 2 mode on
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

### Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the**show etherchannel summary** command:

```
Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
```

```
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

 2      Po2(RU)          -        Te1/0/1(P)  Te1/0/2(P)
```

**Layer 3 EtherChannel Configuration**

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.3 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.4 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
 Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port
```

```
        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

 2      Po2(RU)          -        Te1/0/1(P)  Te1/0/2(P)
```

### Etherchannel Mode — LACP

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode as LACP.

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
Device(config-key-chain)# key 1000
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# channel-group 2 mode active
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/0/2
Device(config-if)# channel-group 2 mode active
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

### Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
```

```
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
 Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1


------+-------------+-----------+-----------------------------------------------

 2      Po2(SU)         LACP      Te1/1/1(P)  Te1/1/2(P)
```

**Layer 3 EtherChannel Configuration**

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.3 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.4 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
 Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
```

```
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

 2      Po2(RU)          LACP      Te1/1/1(P)  Te1/1/2(P)
```

### Etherchannel Mode — PAgP

The following is sample configuration on Device 1 and Device 2 with EtherChannel Mode as PAgP:

```
Device> enable
Device# configure terminal
Device(config)# key chain KC macsec
Device(config-key-chain)# key 1000
Device(config-key-chain)# cryptographic-algorithm aes-128-cmac
Device(config-key-chain)# key-string FC8F5B10557C192F03F60198413D7D45
Device(config-key-chain)# exit
Device(config)# mka policy POLICY
Device(config-mka-policy)# key-server priority 0
Device(config-mka-policy)# macsec-cipher-suite gcm-aes-128
Device(config-mka-policy)# confidentiality-offset 0
Device(config-mka-policy)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# channel-group 2 mode desirable
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/0/2
Device(config-if)# channel-group 2 mode desirable
Device(config-if)# macsec network-link
Device(config-if)# mka policy POLICY
Device(config-if)# mka pre-shared-key key-chain KC
Device(config-if)# end
```

### Layer 2 EtherChannel Configuration

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
```

The following shows a sample output from the **show etherchannel summary** command.

```
 Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1


------+-------------+-----------+-----------------------------------------------

 2      Po2(SU)          PAgP      Te1/1/1(P)  Te1/1/2(P)
```

**Layer 3 EtherChannel Configuration**

Device 1

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.3 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

Device 2

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# no switchport
Device(config-if)# ip address 10.25.25.4 255.255.255.0
Device(config-if)# no shutdown
Device(config-if)# end
```

The following is sample output from the **show etherchannel summary** command:

```
 Flags:  D - down        P - bundled in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
```

```
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator

        M - not in use, minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


 Number of channel-groups in use: 1
 Number of aggregators:           1

 Group  Port-channel  Protocol    Ports

------+-------------+-----------+-----------------------------------------------

 2      Po2(RU)         PAgP      Te1/1/1(P)  Te1/1/2(P)
```

### Displaying Active MKA Sessions

The following shows all the active MKA sessions.

```
Device# show mka sessions interface Te1/0/1

===============================================================================
Interface      Local-TxSCI         Policy-Name      Inherited
Key-Server
Port-ID        Peer-RxSCI          MACsec-Peers     Status           CKN

===============================================================================
Te1/0/1        00a3.d144.3364/0025 POLICY           NO                NO

37             701f.539b.b0c6/0032 1                Secured
1000
```

# Example: Configuring MACsec Cipher Announcement

This example shows how to configure MKA policy for Secure Announcement:

```
Device> enable
Device# configure terminal
Device(config)# mka policy mka_policy
Device(config-mka-policy)# key-server 2
Device(config-mka-policy)# send-secure-announcements
Device(config-mka-policy)#macsec-cipher-suite gcm-aes-128confidentiality-offset 0
Device(config-mka-policy)# end
```

This example shows how to configure Secure Announcement globally:

```
Device> enable
Device# configure terminal
Device(config)# mka defaults policy send-secure-announcements
Device(config)# end
```

This example shows how to configure EAPoL Announcements on an interface:

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 1/0/1
Device(config-if)# eapol announcement
Device(config-if)# end
```

The following is a sample output for **show running-config interface** *interface-name* command with EAPoL announcement enabled.

```
Device# show running-config interface GigabitEthernet 1/0/1

switchport mode access
 macsec
 access-session host-mode multi-host
 access-session closed
 access-session port-control auto
 dot1x pae authenticator
 dot1x timeout quiet-period 10
 dot1x timeout tx-period 5
 dot1x timeout supp-timeout 10
 dot1x supplicant eap profile peap
 eapol announcement
 spanning-tree portfast
 service-policy type control subscriber Dot1X
```

The following is a sample output of the **show mka sessions interface** *interface-name* **detail** command with secure announcement disabled.

```
Device# show mka sessions interface GigabitEthernet 1/0/1 detail


MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec

Local Tx-SCI............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)...........
0100000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89567
EAP Role................. NA
Key Server............... YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (1)
Old SAK Status........... FIRST-SAK
Old SAK AN............... 0
```

```
Old SAK KI (KN)........... FIRST-SAK (0)

SAK Transmit Wait Time... 0s (Not waiting for any peers to respond)
SAK Retire Time.......... 0s (No Old SAK to retire)

MKA Policy Name........... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000001 (GCM-AES-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

# of MACsec Capable Live Peers............ 1
# of MACsec Capable Live Peers Responded.. 1

Live Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89555       c800.8459.e764/002a   10

Potential Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------

Dormant Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
```

The following is sample output of the **show mka sessions details** command with secure announcement disabled.

```
Device# show mka sessions details

MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec

Local Tx-SCI............ 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)...........
0100000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89572
EAP Role................. NA
Key Server............... YES
```

```
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (1)
Old SAK Status........... FIRST-SAK
Old SAK AN............... 0
Old SAK KI (KN).......... FIRST-SAK (0)

SAK Transmit Wait Time... 0s (Not waiting for any peers to respond)
SAK Retire Time.......... 0s (No Old SAK to retire)

MKA Policy Name.......... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000001 (GCM-AES-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

# of MACsec Capable Live Peers............ 1
# of MACsec Capable Live Peers Responded.. 1

Live Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89560       c800.8459.e764/002a   10

Potential Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------

Dormant Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
```

The following is sample output of the **show mka policy** *policy-name* **detail** command with secure announcement disabled.

```
Device# show mka policy p2 detail

MKA Policy Configuration ("p2")
=========================
MKA Policy Name........ p2
Key Server Priority.... 2
Confidentiality Offset. 0
Send Secure Announcement..DISABLED
Cipher Suite(s)........ GCM-AES-128
```

```
Applied Interfaces...
  GigabitEthernet1/0/1
```

# Example: Displaying MKA Information

The following is a sample output from the **show mka sessions** command:

```
Device# show mka sessions

Total MKA Sessions....... 1
      Secured Sessions... 1
      Pending Sessions... 0

==========================================================================================
Interface      Local-TxSCI         Policy-Name      Inherited         Key-Server
Port-ID        Peer-RxSCI          MACsec-Peers     Status            CKN
==========================================================================================
Gi1/0/1        204c.9e85.ede4/002b p2               NO                YES
43             c800.8459.e764/002a 1                Secured
0100000000000000000000000000000000000000000000000000000000000000
```

The following is a sample output from the **show mka sessions interface** *interface-name* command:

```
Device# show mka sessions interface GigabitEthernet 1/0/1

Summary of All Currently Active MKA Sessions on Interface GigabitEthernet1/0/1...

==========================================================================================
Interface      Local-TxSCI         Policy-Name      Inherited         Key-Server
Port-ID        Peer-RxSCI          MACsec-Peers     Status            CKN
==========================================================================================
Gi1/0/1        204c.9e85.ede4/002b p2               NO                YES
43             c800.8459.e764/002a 1                Secured
0100000000000000000000000000000000000000000000000000000000000000
```

The following is sample output from the **show mka sessions interface** *interface-name* **detail** command.

```
Device# show mka sessions interface GigabitEthernet 1/0/1 detail

MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec

Local Tx-SCI............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)........... 0100000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89567
EAP Role................. NA
Key Server............... YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (1)
Old SAK Status........... FIRST-SAK
Old SAK AN............... 0
Old SAK KI (KN).......... FIRST-SAK (0)

SAK Transmit Wait Time... 0s (Not waiting for any peers to respond)
```

```
SAK Retire Time.......... 0s (No Old SAK to retire)

MKA Policy Name.......... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000001 (GCM-AES-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

# of MACsec Capable Live Peers............ 1
# of MACsec Capable Live Peers Responded.. 1

Live Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89555       c800.8459.e764/002a   10

Potential Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------

Dormant Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
```

The following is a sample output from the **show mka sessions details** command:

```
Device# show mka sessions details

MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec

Local Tx-SCI............. 204c.9e85.ede4/002b
Interface MAC Address.... 204c.9e85.ede4
MKA Port Identifier...... 43
Interface Name........... GigabitEthernet1/0/1
Audit Session ID.........
CAK Name (CKN)........... 0100000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)... D46CBEC05D5D67594543CEAE
Message Number (MN)...... 89572
EAP Role................. NA
Key Server............... YES
MKA Cipher Suite......... AES-128-CMAC

Latest SAK Status........ Rx & Tx
Latest SAK AN............ 0
Latest SAK KI (KN)....... D46CBEC05D5D67594543CEAE00000001 (1)
Old SAK Status........... FIRST-SAK
Old SAK AN............... 0
Old SAK KI (KN).......... FIRST-SAK (0)

SAK Transmit Wait Time... 0s (Not waiting for any peers to respond)
SAK Retire Time.......... 0s (No Old SAK to retire)

MKA Policy Name.......... p2
Key Server Priority...... 2
Delay Protection......... NO
Replay Protection........ YES
Replay Window Size....... 0
```

```
Confidentiality Offset... 0
Algorithm Agility........ 80C201
Send Secure Announcement.. DISABLED
SAK Cipher Suite......... 0080C20001000001 (GCM-AES-128)
MACsec Capability........ 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired........... YES

# of MACsec Capable Live Peers............ 1
# of MACsec Capable Live Peers Responded.. 1

Live Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
  38046BA37D7DA77E06D006A9  89560       c800.8459.e764/002a   10

Potential Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------

Dormant Peers List:
  MI                        MN          Rx-SCI (Peer)        KS Priority
  ----------------------------------------------------------------------
```

The following is a sample output from the **show mka policy** command:

```
Device# show mka policy

MKA Policy Summary...

Policy            KS       Delay   Replay  Window Conf   Cipher          Interfaces
Name              Priority Protect Protect Size   Offset Suite(s)        Applied
=============================================================================================
*DEFAULT POLICY*  0        FALSE   TRUE    0      0      GCM-AES-128

p1                1        FALSE   TRUE    0      0      GCM-AES-128

p2                2        FALSE   TRUE    0      0      GCM-AES-128     Gi1/0/1
```

The following is a sample output from the **show mka policy** *policy-name* command:

```
Device# show mka policy p2

MKA Policy Summary...

Policy            KS       Delay   Replay  Window Conf   Cipher          Interfaces
Name              Priority Protect Protect Size   Offset Suite(s)        Applied
=============================================================================================
p2                2        FALSE   TRUE    0      0      GCM-AES-128     Gi1/0/1
```

The following is a sample output from the **show mka policy** *policy-name* **detail** command:

```
Device# show mka policy p2 detail

MKA Policy Configuration ("p2")
========================
MKA Policy Name........ p2
Key Server Priority.... 2
Confidentiality Offset. 0
Send Secure Announcement..DISABLED
Cipher Suite(s)........ GCM-AES-128

Applied Interfaces...
  GigabitEthernet1/0/1
```

The following is a sample output from the **show mka statistics interface** *interface-name* command:

```
Device# show mka statistics interface GigabitEthernet 1/0/1

MKA Statistics for Session
==========================
Reauthentication Attempts.. 0

CA Statistics
   Pairwise CAKs Derived... 0
   Pairwise CAK Rekeys..... 0
   Group CAKs Generated.... 0
   Group CAKs Received..... 0

SA Statistics
   SAKs Generated.......... 1
   SAKs Rekeyed............ 0
   SAKs Received........... 0
   SAK Responses Received.. 1

MKPDU Statistics
   MKPDUs Validated & Rx... 89585
      "Distributed SAK".. 0
      "Distributed CAK".. 0
   MKPDUs Transmitted...... 89596
      "Distributed SAK".. 1
      "Distributed CAK".. 0
```

The following is a sample output from the **show mka summary** command:

```
Device# show mka summary

Total MKA Sessions....... 1
      Secured Sessions... 1
      Pending Sessions... 0

=============================================================================================
Interface      Local-TxSCI         Policy-Name      Inherited         Key-Server
Port-ID        Peer-RxSCI          MACsec-Peers     Status            CKN
=============================================================================================
Gi1/0/1        204c.9e85.ede4/002b p2               NO                YES
43             c800.8459.e764/002a 1                Secured
0100000000000000000000000000000000000000000000000000000000000000



MKA Global Statistics
=====================
MKA Session Totals
   Secured................... 1
   Reauthentication Attempts.. 0

   Deleted (Secured)......... 0
   Keepalive Timeouts........ 0

CA Statistics
   Pairwise CAKs Derived..... 0
   Pairwise CAK Rekeys....... 0
   Group CAKs Generated...... 0
   Group CAKs Received...... 0

SA Statistics
   SAKs Generated............ 1
   SAKs Rekeyed............... 0
   SAKs Received............. 0
   SAK Responses Received..... 1
```

```
MKPDU Statistics
   MKPDUs Validated & Rx...... 89589
      "Distributed SAK"..... 0
      "Distributed CAK"..... 0
   MKPDUs Transmitted......... 89600
      "Distributed SAK"..... 1
      "Distributed CAK"..... 0

MKA Error Counter Totals
========================
Session Failures
   Bring-up Failures................ 0
   Reauthentication Failures........ 0
   Duplicate Auth-Mgr Handle........ 0

SAK Failures
   SAK Generation................... 0
   Hash Key Generation.............. 0
   SAK Encryption/Wrap.............. 0
   SAK Decryption/Unwrap............ 0
   SAK Cipher Mismatch.............. 0

CA Failures
   Group CAK Generation............. 0
   Group CAK Encryption/Wrap........ 0
   Group CAK Decryption/Unwrap...... 0
   Pairwise CAK Derivation.......... 0
   CKN Derivation................... 0
   ICK Derivation................... 0
   KEK Derivation................... 0
   Invalid Peer MACsec Capability... 0
MACsec Failures
   Rx SC Creation................... 0
   Tx SC Creation................... 0
   Rx SA Installation............... 0
   Tx SA Installation............... 0

MKPDU Failures
   MKPDU Tx......................... 0
   MKPDU Rx Validation.............. 0
   MKPDU Rx Bad Peer MN............. 0
   MKPDU Rx Non-recent Peerlist MN.. 0
```

The following is a sample output from the **show macsec interface** command:

```
Device# show macsec interface HundredGigE 2/0/4

 MACsec is enabled
  Replay protect : enabled
  Replay window : 0
  Include SCI : yes
  Use ES Enable : no
  Use SCB Enable : no
  Admin Pt2Pt MAC : forceTrue(1)
  Pt2Pt MAC Operational : no
  Cipher : GCM-AES-128
  Confidentiality Offset : 0

 Capabilities
  ICV length : 16
  Data length change supported: yes
  Max. Rx SA : 16
  Max. Tx SA : 16
  Max. Rx SC : 8
```

```
 Max. Tx SC : 8
 Validate Frames : strict
 PN threshold notification support : Yes
 Ciphers supported : GCM-AES-128
                     GCM-AES-256
                     GCM-AES-XPN-128
                     GCM-AES-XPN-256

Access control : must secure

Transmit Secure Channels
 SCI : 3C5731BBB5850475
 SC state : inUse(1)
  Elapsed time : 7w0d
  Start time : 7w0d
  Current AN: 0
  Previous AN: -
  Next PN: 149757
  SA State: inUse(1)
  Confidentiality : yes
  SAK Unchanged : yes
  SA Create time : 00:04:41
  SA Start time : 7w0d
  SC Statistics
   Auth-only Pkts : 0
   Auth-only Bytes : 0
   Encrypted Pkts : 0
   Encrypted Bytes : 0
  SA Statistics
   Auth-only Pkts : 0
   Auth-only Bytes : 0
   Encrypted Pkts : 149756
   Encrypted Bytes : 16595088

 Port Statistics
  Egress untag pkts  0
  Egress long pkts  0

Receive Secure Channels
 SCI : 3C5731BBB5C504DF
 SC state : inUse(1)
  Elapsed time : 7w0d
  Start time : 7w0d
  Current AN: 0
  Previous AN: -
  Next PN: 149786
  RX SA Count: 0
  SA State: inUse(1)
  SAK Unchanged : yes
  SA Create time : 00:04:39
  SA Start time : 7w0d
  SC Statistics
   Notvalid pkts 0
   Invalid pkts 0
   Valid pkts 0
   Late pkts 0
   Uncheck pkts 0
   Delay pkts 0
   UnusedSA pkts 0
   NousingSA pkts 0
   Validated Bytes 0
   Decrypted Bytes 0
  SA Statistics
   Notvalid pkts 0
```

```
  Invalid pkts 0
  Valid pkts 149784
  Late pkts 0
  Uncheck pkts 0
  Delay pkts 0
  UnusedSA pkts 0
  NousingSA pkts 0
  Validated Bytes 0
  Decrypted Bytes 16654544

Port Statistics
 Ingress untag pkts  0
 Ingress notag pkts  631726
 Ingress badtag pkts  0
 Ingress unknownSCI pkts  0
 Ingress noSCI pkts  0
 Ingress overrun pkts  0
```

# Additional References for MACsec Encryption

### Standards and RFCs

| Standard/RFC | Title |
|---|---|
| IEEE 802.1AE-2006 | *Media Access Control (MAC) Security* |
| IEEE 802.1X-2010 | *Port-Based Network Access Control* |
| IEEE 802.1AEbw-2013 | *Media Access Control (MAC) Security (Amendment to IEEE 802.1AE-2006)—Extended Packet Numbering (XPN)* |
| IEEE 802.1Xbx-2014 | *Port-Based Network Access Control (Amendment to IEEE 802.1X-2010)* |
| RFC 4493 | *The AES-CMAC Algorithm* |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br><br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br><br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/support |

# Feature History for MACsec Encryption

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | MACsec Encryption | MACsec is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. Catalyst switches support 802.1AE encryption with MACsec Key Agreement (MKA) encryption between the switch and host device.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | MACsec Encryption | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Gibraltar 16.12.1 | MKA with High Availability | MKA with high availability is supported.<br><br>Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 14

# Configuring Secure Shell


- Configuring Secure Shell, on page 271


# Configuring Secure Shell

The Secure Shell (SSH) feature is an application and a protocol that provides a secure replacement to the Berkeley r-tools. The protocol secures sessions using standard cryptographic mechanisms, and the application can be used similarly to the Berkeley rexec and rsh tools. Two versions of SSH are available: SSH Version 1 and SSH Version 2. Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only. For information about SSH Version 2, see the “ Secure Shell Version 2 Support” feature module.

## Prerequisites for Configuring Secure Shell

**Note** Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

- For SSH to work, the switch needs an Rivest, Shamir, and Adleman (RSA) public/private key pair. This is the same with Secure Copy Protocol (SCP), which relies on SSH for its secure transport.
- Download the required image on the device. The Secure Shell (SSH) server requires an IPsec (Data Encryption Standard [DES] or 3DES) encryption software image; the SSH client requires an IPsec (DES or 3DES) encryption software image.)
- Configure a hostname and host domain for your device by using the **hostname** and **ip domain name** commands in global configuration mode.
- Generate a Rivest, Shamir, and Adleman (RSA) key pair for your device. This key pair automatically enables SSH and remote authentication when the **crypto key generate rsa** command is entered in global configuration mode.

**Note** To delete the RSA key pair, use the **crypto key zeroize rsa** global configuration command. Once you delete the RSA key pair, you automatically disable the SSH server.

- Configure user authentication for local or remote access. You can configure authentication with or without authentication, authorization, and accounting (AAA).
- The Secure Shell (SSH) server requires an IPsec (Data Encryption Standard [DES] or 3DES) encryption software image; the SSH client requires an IPsec (DES or 3DES) encryption software image.)

# Restrictions for Configuring Secure Shell

**Note** Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

- The Secure Shell (SSH) server and SSH client are supported on Data Encryption Standard (DES) (56-bit) and 3DES (168-bit) data encryption software images only. In DES software images, DES is the only encryption algorithm available. In 3DES software images, both DES and 3DES encryption algorithms are available.
- Execution shell is the only application supported.
- The login banner is not supported in Secure Shell Version 1. It is supported in Secure Shell Version 2.
- The SFTP server is not supported.

# Information About Configuring Secure Shell

Secure Shell (SSH) is a protocol that provides a secure, remote connection to a device. SSH provides more security for remote connections than Telnet does by providing strong encryption when a device is authenticated. This software release supports SSH Version 2 (SSHv2).

## SSH Server

**Note** Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

The Secure Shell (SSH) Server feature enables an SSH client to make a secure, encrypted connection to a Cisco device. This connection provides functionality that is similar to that of an inbound Telnet connection. Before SSH, security was limited to Telnet security. SSH allows a strong encryption to be used with the Cisco software authentication. The SSH server in Cisco software works with publicly and commercially available SSH clients.

## SSH Integrated Client

**Note** Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

The Secure Shell (SSH) Integrated Client feature is an application that runs over the SSH protocol to provide device authentication and encryption. The SSH client enables a Cisco device to make a secure, encrypted connection to another Cisco device or to any other device running the SSH server. This connection provides

functionality similar to that of an outbound Telnet connection except that the connection is encrypted. With authentication and encryption, the SSH client allows for secure communication over an unsecured network.

The SSH client in Cisco software works with publicly and commercially available SSH servers. The SSH client supports the ciphers of Data Encryption Standard (DES), 3DES, and password authentication. User authentication is performed like that in the Telnet session to the device. The user authentication mechanisms supported for SSH are RADIUS, TACACS+, and the use of locally stored usernames and passwords.

**Note** The SSH client functionality is available only when the SSH server is enabled.

## RSA Authentication Support

Rivest, Shamir, and Adleman (RSA) authentication available in Secure Shell (SSH) clients is not supported on the SSH server for Cisco software by default. For more information about RSA authentication support, see the "Configuring a Device for SSH Version 2 Using RSA Pairs" section of the "Secure Shell Version 2 Support" module.

## SSH Servers, Integrated Clients, and Supported Versions

The Secure Shell (SSH) Integrated Client feature is an application that runs over the SSH protocol to provide device authentication and encryption. The SSH client enables a Cisco device to make a secure, encrypted connection to another Cisco device or to any other device running the SSH server. This connection provides functionality similar to that of an outbound Telnet connection except that the connection is encrypted. With authentication and encryption, the SSH client allows for secure communication over an unsecured network.

The SSH server and SSH integrated client are applications that run on the switch. The SSH server works with the SSH client supported in this release and with non-Cisco SSH clients. The SSH client works with publicly and commercially available SSH servers. The SSH client supports the ciphers of Data Encryption Standard (DES), 3DES, and password authentication.

**Note** The SSH client functionality is available only when the SSH server is enabled.

User authentication is performed like that in the Telnet session to the device. SSH also supports the following user authentication methods:

- TACACS+
- RADIUS
- Local authentication and authorization

## SSH Configuration Guidelines

Follow these guidelines when configuring the switch as an SSH server or SSH client:

- An RSA key pair generated by a SSHv1 server can be used by an SSHv2 server, and the reverse.
- If the SSH server is running on an active switch and the active switch fails, the new active switch uses the RSA key pair generated by the previous active switch.

- If you get CLI error messages after entering the **crypto key generate rsa** global configuration command, an RSA key pair has not been generated. Reconfigure the hostname and domain, and then enter the **crypto key generate rsa** command.
- When generating the RSA key pair, the message No host name specified might appear. If it does, you must configure a hostname by using the **hostname** command in global configuration mode.
- When generating the RSA key pair, the message No domain specified might appear. If it does, you must configure an IP domain name by using the **ip domain name** command in global configuration mode.
- When configuring the local authentication and authorization authentication method, make sure that AAA is disabled on the console.

# How to Configure Secure Shell

## Setting Up the Device to Run SSH

Follow the procedure given below to set up your device to run SSH:

### Before you begin

Configure user authentication for local or remote access. This step is required.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **hostname** *hostname*<br>**Example:**<br>`Device(config)# hostname your_hostname` | Configures a hostname and IP domain name for your device.<br>**Note**<br>Follow this procedure only if you are configuring the device as an SSH server. |
| **Step 4** | **ip domain name** *domain_name*<br>**Example:**<br>`Device(config)# ip domain name your_domain` | Configures a host domain for your device. |
| **Step 5** | **crypto key generate rsa**<br>**Example:** | Enables the SSH server for local and remote authentication on the device and generates an |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# crypto key generate rsa` | RSA key pair. Generating an RSA key pair for the device automatically enables SSH.<br>We recommend that a minimum modulus size of 1024 bits.<br>When you generate RSA keys, you are prompted to enter a modulus length. A longer modulus length might be more secure, but it takes longer to generate and to use.<br>**Note**<br>Follow this procedure only if you are configuring the device as an SSH server. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and returns to privileged EXEC mode. |
| **Step 7** | **show ip ssh**<br>**Example:**<br>`Device# show ip ssh` | (Optional) Verifies that the SSH server is enabled and displays the version and configuration data for the SSH connection. |

## Configuring an SSH Server

**Note** Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip ssh** {**time-out** *seconds* \| **authentication-retries** *integer*}<br>**Example:**<br>`Device(config)# ip ssh time-out 30` | Configures Secure Shell (SSH) control parameters.<br>**Note**<br>This command can also be used to establish the number of password prompts provided to |

| | Command or Action | Purpose |
|---|---|---|
| | | the user. The number is the lower of the following two values:<br>• Value proposed by the client using the **ssh -o numberofpasswordprompt** command.<br>• Value configured on the device using the **ip ssh authentication-retries** *integer*command, plus one. |
| **Step 4** | **ip ssh rekey** {**time** *time* \| **volume** *volume*}<br>**Example:**<br>`Device(config)# ip ssh rekey time 108` | (Optional) Configures a time-based rekey or a volume-based rekey for SSH. |
| **Step 5** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show ip ssh**<br>**Example:**<br>`Device# show ip ssh` | (Optional) Verifies that the SSH server is enabled and displays the version and configuration data for the SSH connection. |

## Invoking an SSH Client

**Note** Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

Perform this task to invoke the Secure Shell (SSH) client. The SSH client runs in user EXEC mode and has no specific configuration tasks.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **ssh -l** *username* **-vrf** *vrf-name* *ip-address*<br>**Example:**<br>`Device# ssh -l user1 -vrf vrf1 192.0.2.1` | Invokes the SSH client to connect to an IP host or address in the specified virtual routing and forwarding (VRF) instance. |

# Configuration Examples for Secure Shell

## Example: Configuring an SSH Server

**Note** Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

The following is an example of the Secure Shell (SSH) control parameters configured for the server. In this example, the timeout interval of 30 seconds has been specified. This timeout interval is used during the SSH negotiation phase.

```
Device> enable
Device# configure terminal
Device(config)# ip ssh timeout 30
Device(config)# end
```

## Example: Invoking an SSH Client

**Note** Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

In the following example, the Secure Shell (SSH) client has been invoked to connect to IP address 192.0.2.1 in the specified virtual routing and forwarding (VRF) instance:

```
Device> enable
Device# ssh -1 user1 -vrf vrf1 192.0.2.1
```

## Example: Verifying SSH

**Note** Unless otherwise noted, the term “SSH” denotes “SSH Version 1” only.

To verify that the Secure Shell (SSH) server is enabled and to display the version and configuration data for your SSH connection, use the **show ip ssh** command. The following example shows that SSH is enabled:

```
Device# show ip ssh

SSH Enabled - version 1.5
Authentication timeout: 120 secs; Authentication retries: 3
```

The following example shows that SSH is disabled:

```
Device# show ip ssh

%SSH has not been enabled
```

To verify the status of your SSH server connections, use the **show ssh** command. The following example shows the SSH server connections on the device when SSH is enabled:

```
Device# show ssh

Connection      Version     Encryption State Username
 0 1.5 3DES Session Started  guest
```

The following example shows that SSH is disabled:

```
Device# show ssh

%No SSH server connections running.
```

# Additional References for Secure Shell

### Related Documents

| Related Topic | Document Title |
|---|---|
| SSH Version 2 | Secure Shell Version 2 Support module in the *Security Configuration Guide* |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/techsupport |

# Feature History for Configuring Secure Shell

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Secure Shell | SSH is a protocol that provides a secure, remote connection to a device. SSH provides more security for remote connections than Telnet does by providing strong encryption when a device is authenticated..<br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Secure Shell | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 15

# Secure Shell Version 2 Support

The Secure Shell Version 2 Support feature allows you to configure Secure Shell (SSH) Version 2. (SSH Version 1 support was implemented in an earlier Cisco software release.) SSH runs on top of a reliable transport layer and provides strong authentication and encryption capabilities. The only reliable transport that is defined for SSH is TCP. SSH provides a means to securely access and securely execute commands on another computer over a network. The Secure Copy Protocol (SCP) feature that is provided with SSH allows for the secure transfer of files.


- Prerequisites for Secure Shell Version 2 Support, on page 281
- Restrictions for Secure Shell Version 2 Support, on page 282
- Information About Secure Shell Version 2 Support, on page 282
- How to Configure Secure Shell Version 2 Support, on page 285
- Configuration Examples for Secure Shell Version 2 Support, on page 296
- Additional References for Secure Shell Version 2 Support, on page 300
- Feature History for Secure Shell Version 2 Support, on page 301


## Prerequisites for Secure Shell Version 2 Support

- Before configuring SSH, ensure that the required image is loaded on your device. The SSH server requires you to have a k9 (Triple Data Encryption Standard [3DES]) software image depending on your release.
- You have to use a SSH remote device that supports SSH Version 2 and connect to a Cisco device.
- SCP relies on authentication, authorization, and accounting (AAA) to function correctly. Therefore, AAA must be configured on the device to enable the secure copy protocol on the SSH Server.

**Note** The SSH Version 2 server and the SSH Version 2 client are supported on your Cisco software, depending on your release. (The SSH client runs both the SSH Version 1 protocol and the SSH Version 2 protocol. The SSH client is supported in k9 images depending on your release.)

# Restrictions for Secure Shell Version 2 Support

- Secure Shell (SSH) servers and SSH clients are supported in Triple Data Encryption Standard (3DES) software images.
- Execution Shell, remote command execution, and Secure Copy Protocol (SCP) are the only applications supported.
- Rivest, Shamir, and Adleman (RSA) key generation is an SSH server-side requirement. Devices that act as SSH clients need not generate RSA keys.
- The RSA key pair size must be greater than or equal to 768 bits.
- The following features are not supported:
  - Port forwarding
  - Compression

# Information About Secure Shell Version 2 Support

## Secure Shell Version 2

The Secure Shell Version 2 Support feature allows you to configure SSH Version 2.

The configuration for the SSH Version 2 server is similar to the configuration for SSH Version 1. The **ip ssh version** command defines the SSH version to be configured. If you do not configure this command, SSH by default runs in compatibility mode; that is, both SSH Version 1 and SSH Version 2 connections are honored.

**Note** SSH Version 1 is a protocol that has never been defined in a standard. If you do not want your device to fall back to the undefined protocol (Version 1), you should use the **ip ssh version** command and specify Version 2.

The **ip ssh rsa keypair-name** command enables an SSH connection using the Rivest, Shamir, and Adleman (RSA) keys that you have configured. Previously, SSH was linked to the first RSA keys that were generated (that is, SSH was enabled when the first RSA key pair was generated). This behavior still exists, but by using the **ip ssh rsa keypair-name** command, you can overcome this behavior. If you configure the **ip ssh rsa keypair-name** command with a key pair name, SSH is enabled if the key pair exists or SSH will be enabled if the key pair is generated later. If you use this command to enable SSH, you are not forced to configure a hostname and a domain name, which was required in SSH Version 1 of the Cisco software.

**Note** The login banner is supported in SSH Version 2, but it is not supported in Secure Shell Version 1.

# Secure Shell Version 2 Enhancements

The SSH Version 2 Enhancements feature includes a number of additional capabilities such as supporting Virtual Routing and Forwarding (VRF)-Aware SSH, SSH debug enhancements, and Diffie-Hellman (DH) group exchange support.

**Note** The VRF-Aware SSH feature is supported depending on your release.

The Cisco SSH implementation has traditionally used 768-bit modulus, but with an increasing need for higher key sizes to accommodate DH Group 14 (2048 bits) and Group 16 (4096 bits) cryptographic applications, a message exchange between the client and the server to establish the favored DH group becomes necessary. The **ip ssh dh min size** command configures the modulus size on the SSH server. In addition to this, the **ssh** command was extended to add VRF awareness to the SSH client-side functionality through which the VRF instance name in the client is provided with the IP address to look up the correct routing table and establish a connection.

Debugging was enhanced by modifying SSH debug commands. The **debug ip ssh** command was extended to simplify the debugging process. Before the simplification of the debugging process, this command printed all debug messages related to SSH regardless of what was specifically required. The behavior still exists, but if you configure the **debug ip ssh** command with a keyword, messages are limited to information specified by the keyword.

# Secure Shell Version 2 Enhancements for RSA Keys

Cisco SSH Version 2 supports keyboard-interactive and password-based authentication methods. The SSH Version 2 Enhancements for RSA Keys feature also supports RSA-based public key authentication for the client and the server.

- User authentication: RSA-based user authentication uses a private/public key pair associated with each user for authentication. The user must generate a private/public key pair on the client and configure a public key on the Cisco SSH server to complete the authentication.

  An SSH user trying to establish credentials provides an encrypted signature using the private key. The signature and the user's public key are sent to the SSH server for authentication. The SSH server computes a hash over the public key provided by the user. The hash is used to determine if the server has a matching entry. If a match is found, an RSA-based message verification is performed using the public key. Hence, the user is authenticated or denied access based on the encrypted signature.

- Server authentication: While establishing an SSH session, the Cisco SSH client authenticates the SSH server by using the server host keys available during the key exchange phase. SSH server keys are used to identify the SSH server. These keys are created at the time of enabling SSH and must be configured on the client.

  For server authentication, the Cisco SSH client must assign a host key for each server. When the client tries to establish an SSH session with a server, the client receives the signature of the server as part of the key exchange message. If the strict host key checking flag is enabled on the client, the client checks if it has the host key entry corresponding to the server. If a match is found, the client tries to validate the signature by using the server host key. If the server is successfully authenticated, the session establishment continues; otherwise, it is terminated and displays a "Server Authentication Failed" message.

**Note**

- Storing public keys on a server uses memory; therefore, the number of public keys configurable on an SSH server is restricted to ten users, with a maximum of two public keys per user.
- RSA-based user authentication is supported by the Cisco server, but Cisco clients cannot propose public key as an authentication method. If the Cisco server receives a request from an open SSH client for RSA-based authentication, the server accepts the authentication request.
- For server authentication, configure the RSA public key of the server manually and configure the **ip ssh stricthostkeycheck** command on the Cisco SSH client.

# SSH And Switch Access

Secure Shell (SSH) is a protocol that provides a secure, remote connection to a device. SSH provides more security for remote connections than Telnet does by providing strong encryption when a device is authenticated. This software release supports SSH Version 2 (SSHv2).

SSH functions the same in IPv6 as in IPv4. For IPv6, SSH supports IPv6 addresses and enables secure, encrypted connections with remote IPv6 nodes over an IPv6 transport.

# SNMP Trap Generation

Depending on your release, Simple Network Management Protocol (SNMP) traps are generated automatically when an SSH session terminates if the traps have been enabled and SNMP debugging has been enabled.

**Note** When you configure the **snmp-server host** command, the IP address must be the address of the PC that has the SSH (telnet) client and that has IP connectivity to the SSH server.

You must also enable SNMP debugging using the **debug snmp packet** command to display the traps. The trap information includes information such as the number of bytes sent and the protocol that was used for the SSH session.

# SSH Keyboard Interactive Authentication

The SSH Keyboard Interactive Authentication feature, also known as Generic Message Authentication for SSH, is a method that can be used to implement different types of authentication mechanisms. Basically, any currently supported authentication method that requires only user input can be performed with this feature. The feature is automatically enabled.

The following methods are supported:

- Password
- SecurID and hardware tokens printing a number or a string in response to a challenge sent by the server
- Pluggable Authentication Module (PAM)
- S/KEY (and other One-Time-Pads)

# How to Configure Secure Shell Version 2 Support

## Configuring a Device for SSH Version 2 Using a Hostname and Domain Name

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **hostname** *name*<br>**Example:**<br>`Device(config)# hostname catalyst9k` | Configures a hostname for your device. |
| **Step 4** | **ip domain name** *name*<br>**Example:**<br>`catalyst9k(config)# ip domain name example.com` | Configures a domain name for your device. |
| **Step 5** | **crypto key generate rsa**<br>**Example:**<br>`catalyst9k(config)# crypto key generate rsa` | Enables the SSH server for local and remote authentication. |
| **Step 6** | **ip ssh** [**time-out** *seconds* \| **authentication-retries** *integer*]<br>**Example:**<br>`catalyst9k(config)# ip ssh time-out 120` | (Optional) Configures SSH control variables on your device. |
| **Step 7** | **ip ssh version** [**1** \| **2**]<br>**Example:**<br>`catalyst9k(config)# ip ssh version 1` | (Optional) Specifies the version of SSH to be run on your device. |
| **Step 8** | **exit**<br>**Example:**<br>`catalyst9k(config)# exit` | Exits global configuration mode and enters privileged EXEC mode.<br>• Use **no hostname** command to return to the default host. |

# Configuring a Device for SSH Version 2 Using RSA Key Pairs

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip ssh rsa keypair-name** *keypair-name*<br>**Example:**<br>`Device(config)# ip ssh rsa keypair-name sshkeys` | Specifies the RSA key pair to be used for SSH.<br>**Note**<br>A Cisco device can have many RSA key pairs. |
| **Step 4** | **crypto key generate rsa usage-keys label** *key-label* **modulus** *modulus-size*<br>**Example:**<br>`Device(config)# crypto key generate rsa usage-keys label sshkeys modulus 768` | Enables the SSH server for local and remote authentication on the device.<br>• For SSH Version 2, the modulus size must be at least 768 bits.<br>**Note**<br>To delete the RSA key pair, use the **crypto key zeroize rsa** command. When you delete the RSA key pair, you automatically disable the SSH server. |
| **Step 5** | **ip ssh** [**time-out** *seconds* \| **authentication-retries** *integer*]<br>**Example:**<br>`Device(config)# ip ssh time-out 12` | Configures SSH control variables on your device. |
| **Step 6** | **ip ssh version 2**<br>**Example:**<br>`Device(config)# ip ssh version 2` | Specifies the version of SSH to be run on the device. |
| **Step 7** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and enters privileged EXEC mode. |

# Configuring the Cisco SSH Server to Perform RSA-Based User Authentication

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **hostname** *name*<br>**Example:**<br>`Device(config)# hostname host1` | Specifies the hostname. |
| **Step 4** | **ip domain name** *name*<br>**Example:**<br>`host1(config)# ip domain name name1` | Defines a default domain name that the Cisco software uses to complete unqualified hostnames. |
| **Step 5** | **crypto key generate rsa**<br>**Example:**<br>`host1(config)# crypto key generate rsa` | Generates RSA key pairs. |
| **Step 6** | **ip ssh pubkey-chain**<br>**Example:**<br>`host1(config)# ip ssh pubkey-chain` | Configures SSH-RSA keys for user and server authentication on the SSH server and enters public-key configuration mode.<br>• The user authentication is successful if the RSA public key stored on the server is verified with the public or the private key pair stored on the client. |
| **Step 7** | **username** *username*<br>**Example:**<br>`host1(conf-ssh-pubkey)# username user1` | Configures the SSH username and enters public-key user configuration mode. |
| **Step 8** | **key-string**<br>**Example:**<br>`host1(conf-ssh-pubkey-user)# key-string` | Specifies the RSA public key of the remote peer and enters public-key data configuration mode.<br>**Note**<br>You can obtain the public key value from an open SSH client; that is, from the .ssh/id_rsa.pub file. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **key-hash** *key-type* *key-name*<br>**Example:**<br>`host1(conf-ssh-pubkey-data)# key-hash ssh-rsa key1` | (Optional) Specifies the SSH key type and version.<br>• The key type must be ssh-rsa for the configuration of private public key pairs.<br>• This step is optional only if the **key-string** command is configured.<br>• You must configure either the **key-string** command or the **key-hash** command.<br>**Note**<br>You can use a hashing software to compute the hash of the public key string, or you can also copy the hash value from another Cisco device. Entering the public key data using the **key-string** command is the preferred way to enter the public key data for the first time. |
| **Step 10** | **end**<br>**Example:**<br>`host1(conf-ssh-pubkey-data)# end` | Exits public-key data configuration mode and returns to privileged EXEC mode.<br>• Use **no hostname** command to return to the default host. |

# Configuring the Cisco IOS SSH Client to Perform RSA-Based Server Authentication

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **hostname** *name*<br>**Example:**<br>`Device(config)# hostname host1` | Specifies the hostname. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **ip domain name** *name*<br>**Example:**<br>`host1(config)# ip domain name name1` | Defines a default domain name that the Cisco software uses to complete unqualified hostnames. |
| **Step 5** | **crypto key generate rsa**<br>**Example:**<br>`host1(config)# crypto key generate rsa` | Generates RSA key pairs. |
| **Step 6** | **ip ssh pubkey-chain**<br>**Example:**<br>`host1(config)# ip ssh pubkey-chain` | Configures SSH-RSA keys for user and server authentication on the SSH server and enters public-key configuration mode. |
| **Step 7** | **server** *server-name*<br>**Example:**<br>`host1(conf-ssh-pubkey)# server server1` | Enables the SSH server for public-key authentication on the device and enters public-key server configuration mode. |
| **Step 8** | **key-string**<br>**Example:**<br>`host1(conf-ssh-pubkey-server)# key-string` | Specifies the RSA public-key of the remote peer and enters public key data configuration mode.<br>**Note**<br>You can obtain the public key value from an open SSH client; that is, from the .ssh/id_rsa.pub file. |
| **Step 9** | **exit**<br>**Example:**<br>`host1(conf-ssh-pubkey-data)# exit` | Exits public-key data configuration mode and enters public-key server configuration mode. |
| **Step 10** | **key-hash** *key-type* *key-name*<br>**Example:**<br>`host1(conf-ssh-pubkey-server)# key-hash ssh-rsa key1` | (Optional) Specifies the SSH key type and version.<br>• The key type must be ssh-rsa for the configuration of private/public key pairs.<br>• This step is optional only if the **key-string** command is configured.<br>• You must configure either the **key-string** command or the **key-hash** command.<br>**Note**<br>You can use a hashing software to compute the hash of the public key string, or you can copy the hash value from another Cisco device. Entering the public key data using the **key-string** command is the preferred way to enter the public key data for the first time. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 11** | **end**<br>**Example:**<br>`host1(conf-ssh-pubkey-server)# end` | Exits public-key server configuration mode and returns to privileged EXEC mode. |
| **Step 12** | **configure terminal**<br>**Example:**<br>`host1# configure terminal` | Enters global configuration mode. |
| **Step 13** | **ip ssh stricthostkeycheck**<br>**Example:**<br>`host1(config)# ip ssh stricthostkeycheck` | Ensures that server authentication takes place.<br>• The connection is terminated in case of a failure.<br>• Use **no hostname** command to return to the default host. |
| **Step 14** | **end**<br>**Example:**<br>`host1(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Starting an Encrypted Session with a Remote Device

**Note** The device with which you want to connect must support a Secure Shell (SSH) server that has an encryption algorithm that is supported in Cisco software. Also, you need not enable your device. SSH can be run in disabled mode.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **ssh** [**-v** {**1** \| **2**} \| **-c** {**aes128-ctr** \| **aes192-ctr** \| **aes256-ctr** \| **aes128-cbc** \| **3des** \| **aes192-cbc** \| **aes256-cbc**} \| **-l** *user-id* \| **-l** *user-id*:*vrf-name number ip-address ip-address* \| **-l** *user-id*:*rotary number ip-address* \| **-m** {**hmac-md5-128** \| **hmac-md5-96** \| **hmac-sha1-160** \| **hmac-sha1-96**} \| **-o numberofpasswordprompts** *n* \| **-p** *port-num*] {*ip-addr* \| *hostname*} [**command** \| **-vrf**] | Starts an encrypted session with a remote networking device. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device# ssh -v 2 -c aes256-ctr -m hmac-sha1-96 -l user2 10.76.82.24` | |

# Verifying the Status of the Secure Shell Connection

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **show ssh**<br>**Example:**<br>`Device# show ssh` | Displays the status of SSH server connections. |
| **Step 3** | **exit**<br>**Example:**<br>`Device# exit` | Exits privileged EXEC mode and returns to user EXEC mode. |

The following sample output from the **show ssh** command displays status of various SSH Version 1 and Version 2 connections for Version 1 and Version 2 connections:

```
---------------------------------------------------------------------
Device# show ssh

Connection      Version Encryption      State                   Username
 0              1.5     3DES            Session started         lab
Connection Version Mode Encryption  Hmac                State
Username
1          2.0     IN   aes128-cbc  hmac-md5     Session started       lab
1          2.0     OUT  aes128-cbc  hmac-md5     Session started       lab
-------------------------------------------------------------------------
```

The following sample output from the **show ssh** command displays status of various SSH Version 1 and Version 2 connections for a Version 2 connection with no Version 1 connection:

```
--------------------------------------------------------------------------
Device# show ssh

Connection Version Mode Encryption  Hmac                State
Username
1          2.0     IN   aes128-cbc  hmac-md5     Session started       lab
1          2.0     OUT  aes128-cbc  hmac-md5     Session started       lab
%No SSHv1 server connections running.
--------------------------------------------------------------------------
```

The following sample output from the **show ssh** command displays status of various SSH Version 1 and Version 2 connections for a Version 1 connection with no Version 2 connection:

```
-----------------------------------------------------------------------
Device# show ssh

Connection      Version Encryption      State                   Username
 0              1.5     3DES            Session started         lab
%No SSHv2 server connections running.
-----------------------------------------------------------------------
```

# Verifying the Secure Shell Version 2 Status

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **show ip ssh**<br>**Example:**<br>`Device# show ip ssh` | Displays the version and configuration data for SSH. |
| **Step 3** | **exit**<br>**Example:**<br>`Device# exit` | Exits privileged EXEC mode and returns to user EXEC mode. |

### Examples

The following sample output from the **show ip ssh** command displays the version of SSH that is enabled, the authentication timeout values, and the number of authentication retries for Version 1 and Version 2 connections:

```
-----------------------------------------------------------------------
Device# show ip ssh

SSH Enabled - version 1.99
Authentication timeout: 120 secs; Authentication retries: 3
-----------------------------------------------------------------------
```

The following sample output from the **show ip ssh** command displays the version of SSH that is enabled, the authentication timeout values, and the number of authentication retries for a Version 2 connection with no Version 1 connection:

```
-----------------------------------------------------------------------
Device# show ip ssh

SSH Enabled - version 2.0
Authentication timeout: 120 secs; Authentication retries: 3
-----------------------------------------------------------------------
```

The following sample output from the **show ip ssh** command displays the version of SSH that is enabled, the authentication timeout values, and the number of authentication retries for a Version 1 connection with no Version 2 connection:

```
----------------------------------------------------------------------
Device# show ip ssh

3d06h: %SYS-5-CONFIG_I: Configured from console by console
SSH Enabled - version 1.5
Authentication timeout: 120 secs; Authentication retries: 3
----------------------------------------------------------------------
```

# Monitoring and Maintaining Secure Shell Version 2

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **debug ip ssh**<br>**Example:**<br>`Device# debug ip ssh` | Enables debugging of SSH. |
| **Step 3** | **debug snmp packet**<br>**Example:**<br>`Device# debug snmp packet` | Enables debugging of every SNMP packet sent or received by the device. |

### Example

The following sample output from the **debug ip ssh** command shows the connection is an SSH Version 2 connection:

```
Device# debug ip ssh

00:33:55: SSH1: starting SSH control process
00:33:55: SSH1: sent protocol version id SSH-1.99-Cisco-1.25
00:33:55: SSH1: protocol version id is - SSH-2.0-OpenSSH_2.5.2p2
00:33:55: SSH2 1: send: len 280 (includes padlen 4)
00:33:55: SSH2 1: SSH2_MSG_KEXINIT sent
00:33:55: SSH2 1: ssh_receive: 536 bytes received
00:33:55: SSH2 1: input: packet len 632
00:33:55: SSH2 1: partial packet 8, need 624, maclen 0
00:33:55: SSH2 1: ssh_receive: 96 bytes received
00:33:55: SSH2 1: partial packet 8, need 624, maclen 0
00:33:55: SSH2 1: input: padlen 11
00:33:55: SSH2 1: received packet type 20
00:33:55: SSH2 1: SSH2_MSG_KEXINIT received
00:33:55: SSH2: kex: client->server aes128-cbc hmac-md5 none
00:33:55: SSH2: kex: server->client aes128-cbc hmac-md5 none
00:33:55: SSH2 1: expecting SSH2_MSG_KEXDH_INIT
```

```
00:33:55: SSH2 1: ssh_receive: 144 bytes received
00:33:55: SSH2 1: input: packet len 144
00:33:55: SSH2 1: partial packet 8, need 136, maclen 0
00:33:55: SSH2 1: input: padlen 5
00:33:55: SSH2 1: received packet type 30
00:33:55: SSH2 1: SSH2_MSG_KEXDH_INIT received
00:33:55: SSH2 1: signature length 111
00:33:55: SSH2 1: send: len 384 (includes padlen 7)
00:33:55: SSH2: kex_derive_keys complete
00:33:55: SSH2 1: send: len 16 (includes padlen 10)
00:33:55: SSH2 1: newkeys: mode 1
00:33:55: SSH2 1: SSH2_MSG_NEWKEYS sent
00:33:55: SSH2 1: waiting for SSH2_MSG_NEWKEYS
00:33:55: SSH2 1: ssh_receive: 16 bytes received
00:33:55: SSH2 1: input: packet len 16
00:33:55: SSH2 1: partial packet 8, need 8, maclen 0
00:33:55: SSH2 1: input: padlen 10
00:33:55: SSH2 1: newkeys: mode 0
00:33:55: SSH2 1: received packet type 2100:33:55: SSH2 1: SSH2_MSG_NEWKEYS received
00:33:56: SSH2 1: ssh_receive: 48 bytes received
00:33:56: SSH2 1: input: packet len 32
00:33:56: SSH2 1: partial packet 16, need 16, maclen 16
00:33:56: SSH2 1: MAC #3 ok
00:33:56: SSH2 1: input: padlen 10
00:33:56: SSH2 1: received packet type 5
00:33:56: SSH2 1: send: len 32 (includes padlen 10)
00:33:56: SSH2 1: done calc MAC out #3
00:33:56: SSH2 1: ssh_receive: 64 bytes received
00:33:56: SSH2 1: input: packet len 48
00:33:56: SSH2 1: partial packet 16, need 32, maclen 16
00:33:56: SSH2 1: MAC #4 ok
00:33:56: SSH2 1: input: padlen 9
00:33:56: SSH2 1: received packet type 50
00:33:56: SSH2 1: send: len 32 (includes padlen 13)
00:33:56: SSH2 1: done calc MAC out #4
00:34:04: SSH2 1: ssh_receive: 160 bytes received
00:34:04: SSH2 1: input: packet len 64
00:34:04: SSH2 1: partial packet 16, need 48, maclen 16
00:34:04: SSH2 1: MAC #5 ok
00:34:04: SSH2 1: input: padlen 13
00:34:04: SSH2 1: received packet type 50
00:34:04: SSH2 1: send: len 16 (includes padlen 10)
00:34:04: SSH2 1: done calc MAC out #5
00:34:04: SSH2 1: authentication successful for lab
00:34:04: SSH2 1: input: packet len 64
00:34:04: SSH2 1: partial packet 16, need 48, maclen 16
00:34:04: SSH2 1: MAC #6 ok
00:34:04: SSH2 1: input: padlen 6
00:34:04: SSH2 1: received packet type 2
00:34:04: SSH2 1: ssh_receive: 64 bytes received
00:34:04: SSH2 1: input: packet len 48
00:34:04: SSH2 1: partial packet 16, need 32, maclen 16
00:34:04: SSH2 1: MAC #7 ok
00:34:04: SSH2 1: input: padlen 19
00:34:04: SSH2 1: received packet type 90
00:34:04: SSH2 1: channel open request
00:34:04: SSH2 1: send: len 32 (includes padlen 10)
00:34:04: SSH2 1: done calc MAC out #6
00:34:04: SSH2 1: ssh_receive: 192 bytes received
00:34:04: SSH2 1: input: packet len 64
00:34:04: SSH2 1: partial packet 16, need 48, maclen 16
00:34:04: SSH2 1: MAC #8 ok
00:34:04: SSH2 1: input: padlen 13
00:34:04: SSH2 1: received packet type 98
```

```
00:34:04: SSH2 1: pty-req request
00:34:04: SSH2 1: setting TTY - requested: height 24, width 80; set: height 24,
width 80
00:34:04: SSH2 1: input: packet len 96
00:34:04: SSH2 1: partial packet 16, need 80, maclen 16
00:34:04: SSH2 1: MAC #9 ok
00:34:04: SSH2 1: input: padlen 11
00:34:04: SSH2 1: received packet type 98
00:34:04: SSH2 1: x11-req request
00:34:04: SSH2 1: ssh_receive: 48 bytes received
00:34:04: SSH2 1: input: packet len 32
00:34:04: SSH2 1: partial packet 16, need 16, maclen 16
00:34:04: SSH2 1: MAC #10 ok
00:34:04: SSH2 1: input: padlen 12
00:34:04: SSH2 1: received packet type 98
00:34:04: SSH2 1: shell request
00:34:04: SSH2 1: shell message received
00:34:04: SSH2 1: starting shell for vty
00:34:04: SSH2 1: send: len 48 (includes padlen 18)
00:34:04: SSH2 1: done calc MAC out #7
00:34:07: SSH2 1: ssh_receive: 48 bytes received
00:34:07: SSH2 1: input: packet len 32
00:34:07: SSH2 1: partial packet 16, need 16, maclen 16
00:34:07: SSH2 1: MAC #11 ok
00:34:07: SSH2 1: input: padlen 17
00:34:07: SSH2 1: received packet type 94
00:34:07: SSH2 1: send: len 32 (includes padlen 17)
00:34:07: SSH2 1: done calc MAC out #8
00:34:07: SSH2 1: ssh_receive: 48 bytes received
00:34:07: SSH2 1: input: packet len 32
00:34:07: SSH2 1: partial packet 16, need 16, maclen 16
00:34:07: SSH2 1: MAC #12 ok
00:34:07: SSH2 1: input: padlen 17
00:34:07: SSH2 1: received packet type 94
00:34:07: SSH2 1: send: len 32 (includes padlen 17)
00:34:07: SSH2 1: done calc MAC out #9
00:34:07: SSH2 1: ssh_receive: 48 bytes received
00:34:07: SSH2 1: input: packet len 32
00:34:07: SSH2 1: partial packet 16, need 16, maclen 16
00:34:07: SSH2 1: MAC #13 ok
00:34:07: SSH2 1: input: padlen 17
00:34:07: SSH2 1: received packet type 94
00:34:07: SSH2 1: send: len 32 (includes padlen 17)
00:34:07: SSH2 1: done calc MAC out #10
00:34:08: SSH2 1: ssh_receive: 48 bytes received
00:34:08: SSH2 1: input: packet len 32
00:34:08: SSH2 1: partial packet 16, need 16, maclen 16
00:34:08: SSH2 1: MAC #14 ok
00:34:08: SSH2 1: input: padlen 17
00:34:08: SSH2 1: received packet type 94
00:34:08: SSH2 1: send: len 32 (includes padlen 17)
00:34:08: SSH2 1: done calc MAC out #11
00:34:08: SSH2 1: ssh_receive: 48 bytes received
00:34:08: SSH2 1: input: packet len 32
00:34:08: SSH2 1: partial packet 16, need 16, maclen 16
00:34:08: SSH2 1: MAC #15 ok
00:34:08: SSH2 1: input: padlen 17
00:34:08: SSH2 1: received packet type 94
00:34:08: SSH2 1: send: len 32 (includes padlen 16)
00:34:08: SSH2 1: done calc MAC out #12
00:34:08: SSH2 1: send: len 48 (includes padlen 18)
00:34:08: SSH2 1: done calc MAC out #13
00:34:08: SSH2 1: send: len 16 (includes padlen 6)
00:34:08: SSH2 1: done calc MAC out #14
```

```
00:34:08: SSH2 1: send: len 16 (includes padlen 6)
00:34:08: SSH2 1: done calc MAC out #15
00:34:08: SSH1: Session terminated normally
```

# Configuration Examples for Secure Shell Version 2 Support

## Example: Configuring Secure Shell Version 2

```
Device> enable
Device# configure terminal
Device(config)# ip ssh version 2
Device(config)# end
```

## Example: Configuring Secure Shell Versions 1 and 2

```
Device> enable
Device# configure terminal
Device(config)# no ip ssh version
Device(config)# end
```

## Example: Starting an Encrypted Session with a Remote Device

```
Device> enable
Device# ssh -v 2 -c aes256-cbc -m hmac-sha1-160 -l shaship 10.76.82.24
Device# exit
```

## Example: Setting an SNMP Trap

The following example shows how to set an SNMP trap is set. The trap notification is generated automatically when the SSH session terminates. In the example, 10.1.1.1 is the IP address of the SSH client.

```
Device> enable
Device# configure terminal
Device(config)# snmp-server trap link switchover
Device(config)# snmp-server host 10.1.1.1 public tty
Device(config)# end
```

## Examples: SSH Keyboard Interactive Authentication

### Example: Enabling Client-Side Debugs

The following example shows that the client-side debugs are turned on, and the maximum number of prompts is six (three for the SSH keyboard interactive authentication method and three for the password authentication method).

```
Password:
Password:
Password:
Password:
Password:
Password: cisco123
```

```
Last login: Tue Dec 6 13:15:21 2005 from 10.76.248.213
user1@courier:~> exit
logout
[Connection to 10.76.248.200 closed by foreign host]
Device1# debug ip ssh client

SSH Client debugging is on

Device1# ssh -l lab 10.1.1.3

Password:
*Nov 17 12:50:53.199: SSH0: sent protocol version id SSH-1.99-Cisco-1.25
*Nov 17 12:50:53.199: SSH CLIENT0: protocol version id is - SSH-1.99-Cisco-1.25
*Nov 17 12:50:53.199: SSH CLIENT0: sent protocol version id SSH-1.99-Cisco-1.25
*Nov 17 12:50:53.199: SSH CLIENT0: protocol version exchange successful
*Nov 17 12:50:53.203: SSH0: protocol version id is - SSH-1.99-Cisco-1.25
*Nov 17 12:50:53.335: SSH CLIENT0: key exchange successful and encryption on
*Nov 17 12:50:53.335: SSH2 CLIENT 0: using method keyboard-interactive
Password:
Password:
Password:
*Nov 17 12:51:01.887: SSH2 CLIENT 0: using method password authentication
Password:
Password: lab
Device2>

*Nov 17 12:51:11.407: SSH2 CLIENT 0: SSH2_MSG_USERAUTH_SUCCESS message received
*Nov 17 12:51:11.407: SSH CLIENT0: user authenticated
*Nov 17 12:51:11.407: SSH2 CLIENT 0: pty-req request sent
*Nov 17 12:51:11.411: SSH2 CLIENT 0: shell request sent
*Nov 17 12:51:11.411: SSH CLIENT0: session open
```

## Example: Enabling ChPass with a Blank Password Change

In the following example, the ChPass feature is enabled, and a blank password change is accomplished using the SSH Keyboard Interactive Authentication method. A TACACS+ access control server (ACS) is used as the back-end AAA server.

```
Device> enable
Device1# ssh -l cisco 10.1.1.3

Password:
Old Password: cisco
New Password: cisco123
Re-enter New password: cisco123

Device2> exit

[Connection to 10.1.1.3 closed by foreign host]
```

## Example: Enabling ChPass and Changing the Password on First Login

In the following example, the ChPass feature is enabled and TACACS+ ACS is used as the back-end server. The password is changed on the first login using the SSH keyboard interactive authentication method.

```
Device1> enable
Device1# ssh -l cisco 10.1.1.3

Password: cisco
Your password has expired.
Enter a new one now.
New Password: cisco123
```

```
Re-enter New password: cisco123

Device2> exit

[Connection to 10.1.1.3 closed by foreign host]

Device1# ssh -l cisco 10.1.1.3

Password:cisco1
Your password has expired.
Enter a new one now.
New Password: cisco
Re-enter New password: cisco12
The New and Re-entered passwords have to be the same.
Try again.
New Password: cisco
Re-enter New password: cisco

Device2>
```

## Example: Enabling ChPass and Expiring the Password After Three Logins

In the following example, the ChPass feature is enabled and TACACS+ ACS is used as the back-end AAA server. The password expires after three logins using the SSH keyboard interactive authentication method.

```
Device# ssh -l cisco. 10.1.1.3

Password: cisco

Device2> exit

[Connection to 10.1.1.3 closed by foreign host]

Device1# ssh -l cisco 10.1.1.3

Password: cisco

Device2> exit

Device1# ssh -l cisco 10.1.1.3

Password: cisco

Device2> exit

[Connection to 10.1.1.3 closed by foreign host]

Device1# ssh -l cisco 10.1.1.3

Password: cisco
Your password has expired.
Enter a new one now.
New Password: cisco123
Re-enter New password: cisco123

Device2>
```

# Example: SNMP Debugging

The following is sample output from the **debug snmp packet** command. The output provides SNMP trap information for an SSH session.

```
Device1# debug snmp packet

SNMP packet debugging is on
Device1# ssh -l lab 10.0.0.2
Password:

Device2# exit

[Connection to 10.0.0.2 closed by foreign host]
Device1#
*Jul 18 10:18:42.619: SNMP: Queuing packet to 10.0.0.2
*Jul 18 10:18:42.619: SNMP: V1 Trap, ent cisco, addr 10.0.0.1, gentrap 6, spectrap 1
local.9.3.1.1.2.1 = 6
tcpConnEntry.1.10.0.0.1.22.10.0.0.2.55246 = 4
ltcpConnEntry.5.10.0.0.1.22.10.0.0.2.55246 = 1015
ltcpConnEntry.1.10.0.0.1.22.10.0.0.2.55246 = 1056
ltcpConnEntry.2.10.0.0.1.22.10.0.0.2.55246 = 1392
local.9.2.1.18.2 = lab
*Jul 18 10:18:42.879: SNMP: Packet sent via UDP to 10.0.0.2

Device1#
```

# Examples: SSH Debugging Enhancements

The following is sample output from the **debug ip ssh detail** command. The output provides debugging information about the SSH protocol and channel requests.

```
Device# debug ip ssh detail

00:04:22: SSH0: starting SSH control process
00:04:22: SSH0: sent protocol version id SSH-1.99-Cisco-1.25
00:04:22: SSH0: protocol version id is - SSH-1.99-Cisco-1.25
00:04:22: SSH2 0: SSH2_MSG_KEXINIT sent
00:04:22: SSH2 0: SSH2_MSG_KEXINIT received
00:04:22: SSH2:kex: client->server enc:aes128-cbc mac:hmac-sha1
00:04:22: SSH2:kex: server->client enc:aes128-cbc mac:hmac-sha1
00:04:22: SSH2 0: expecting SSH2_MSG_KEXDH_INIT
00:04:22: SSH2 0: SSH2_MSG_KEXDH_INIT received
00:04:22: SSH2: kex_derive_keys complete
00:04:22: SSH2 0: SSH2_MSG_NEWKEYS sent
00:04:22: SSH2 0: waiting for SSH2_MSG_NEWKEYS
00:04:22: SSH2 0: SSH2_MSG_NEWKEYS received
00:04:24: SSH2 0: authentication successful for lab
00:04:24: SSH2 0: channel open request
00:04:24: SSH2 0: pty-req request
00:04:24: SSH2 0: setting TTY - requested: height 24, width 80; set: height 24, width 80
00:04:24: SSH2 0: shell request
00:04:24: SSH2 0: shell message received
00:04:24: SSH2 0: starting shell for vty
00:04:38: SSH0: Session terminated normally
```

The following is sample output from the **debug ip ssh packet** command. The output provides debugging information about the SSH packet.

```
Device# debug ip ssh packet

00:05:43: SSH2 0: send:packet of  length 280 (length also includes padlen of 4)
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: input: total packet length of 280 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
```

```
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 24 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 272 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 4 bytes
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: input: total packet length of 144 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 64 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: ssh_receive: 16 bytes received
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 136 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 6 bytes
00:05:43: SSH2 0: signature length 143
00:05:43: SSH2 0: send:packet of  length 448 (length also includes padlen of 7)
00:05:43: SSH2 0: send:packet of  length 16 (length also includes padlen of 10)
00:05:43: SSH2 0: newkeys: mode 1
00:05:43: SSH2 0: ssh_receive: 16 bytes received
00:05:43: SSH2 0: input: total packet length of 16 bytes
00:05:43: SSH2 0: partial packet length(block size)8 bytes,needed 8 bytes, maclen 0
00:05:43: SSH2 0: input: padlength 10 bytes
00:05:43: SSH2 0: newkeys: mode 0
00:05:43: SSH2 0: ssh_receive: 52 bytes received
00:05:43: SSH2 0: input: total packet length of 32 bytes
00:05:43: SSH2 0: partial packet length(block size)16 bytes,needed 16 bytes, maclen 20
00:05:43: SSH2 0: MAC compared for #3 :ok
```

# Additional References for Secure Shell Version 2 Support

### Related Documents

| Related Topic | Document Title |
|---|---|
| SSH Version 1 | Configuring Secure Shell chapter of the *Security Configuration Guide* |

### Standards

| Standards | Title |
|---|---|
| IETF Secure Shell Version 2 Draft Standards | Internet Engineering Task Force website |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br><br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br><br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/support |

# Feature History for Secure Shell Version 2 Support

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Secure Shell Version 2 Support | The Secure Shell Version 2 Support feature allows you to configure Secure Shell (SSH) Version 2 (SSH Version 1 support was implemented in an earlier Cisco IOS software release). SSH runs on top of a reliable transport layer and provides strong authentication and encryption capabilities. SSH version 2 also supports AES counter-based encryption mode.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Secure Shell Version 2 Support | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 16

# SSH Support Over IPv6

Secure Shell (SSH) provides support for IPv6 addresses that enable a Cisco device to accept and establish secure, encrypted connections with remote IPv6 nodes over an IPv6 transport.


- Prerequisites for SSH Support over IPv6, on page 303
- Information About SSH Support over IPv6, on page 303
- How to Configure SSH Support over IPv6, on page 304
- Configuration Examples for SSH Support over IPv6, on page 305
- Additional References for SSH Support over IPv6, on page 305
- Feature History for SSH Support over IPv6, on page 305


# Prerequisites for SSH Support over IPv6

- An IPsec (Data Encryption Standard [DES] or 3DES) encryption software image is loaded on your device. IPv6 transport for the SSH server and SSH client requires an IPsec encryption software image.
- A hostname and host domain are configured for your device.
- A Rivest, Shamir, and Adelman (RSA) key pair, which automatically enables SSH, is generated for your device.
- A user authentication mechanism for local or remote access is configured on your device.
- To authenticate SSH clients, configure TACACS+ or RADIUS over an IPv4 transport and then connect to an SSH server over an IPv6 transport.

The basic restrictions for SSH over an IPv4 transport apply to SSH over an IPv6 transport. The use of locally stored usernames and passwords is the only user authentication mechanism supported by SSH over an IPv6 transport. TACACS+ and RADIUS user authentication mechanisms are not supported over an IPv6 transport.

# Information About SSH Support over IPv6

## SSH over an IPv6 Transport

Secure shell (SSH) SSH in IPv6 functions the same and offers the same benefits as SSH in IPv4. The SSH server feature enables an SSH client to make a secure, encrypted connection to a Cisco device, and the SSH

client feature enables a Cisco device to make a secure, encrypted connection to another Cisco device or to any other device running an SSH server. IPv6 enhancements to SSH consist of support for IPv6 addresses that enable a Cisco device to accept and establish secure, encrypted connections with remote IPv6 nodes over an IPv6 transport.

# How to Configure SSH Support over IPv6

## Enabling SSH on an IPv6 Device

This task is optional. If you do not configure SSH parameters, then the default values will be used.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip ssh** [**timeout** *seconds* \| **authentication-retries** *integer*]<br>**Example:**<br>`Device(config)# IP ssh timeout 100 authentication-retries 2` | Configures SSH control variables on your device. |
| **Step 4** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode, and returns to privileged EXEC mode. |
| **Step 5** | **ssh** [ **-v** { **1** \| **2** } \| **c** { **3des** \| **aes128-cbc** \| **aes192-cbc** \| **aes256-cbc** } \| **-l** *userid* \| **-l** *userid:vrfname number ip-address ip-address* \| **-l** *userid:rotary number ip-address* \| **-m** { **hmac-md5** \| **hmac-md5-96** \| **hmac-sha1** \| **hmac-sha1-96** } \| **-o numberofpasswordprompts** *n* \| **-p** *port-num*] { **ip-addr** \| **hostname**} [ **command** \| **-vrf**]<br>**Example:**<br>`Device# ssh -l userid1 2001:db8:2222:1044::72` | Starts an encrypted session with a remote networking device. |

# Configuration Examples for SSH Support over IPv6

## Example: Enabling SSH on an IPv6 Device

```
Device# configure terminal
Device(config)# ip ssh
Device(config)# exit
Device# ssh -l userid1 2001:db8:2222:1044::72
```

# Additional References for SSH Support over IPv6

### Related Documents

| Related Topic | Document Title |
|---|---|
| SSH Version 1 and Version 2 Support | Configuring Secure Shell and Secure Shell Version 2 Support chapters of the *Security Configuration Guide.* |

### Standards and RFCs

| Standard/RFC | Title |
|---|---|
| RFCs for IPv6 | *IPv6 RFCs* |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/support |

# Feature History for SSH Support over IPv6

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | SSH Support over IPv6 | SSH provides support for IPv6 addresses that enable a Cisco device to accept and establish secure, encrypted connections with remote IPv6 nodes over an IPv6 transport.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | SSH Support over IPv6 | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 17

# Configuring SSH File Transfer Protocol

Secure Shell (SSH) includes support for SSH File Transfer Protocol (SFTP), which is a new standard file transfer protocol introduced in SSHv2. This feature provides a secure and authenticated method for copying device configuration or device image files.


- Prerequisites for SSH File Transfer Protocol, on page 307
- Restrictions for SSH File Transfer Protocol, on page 307
- Information About SSH Support over IPv6, on page 308
- How to Configure SSH File Transfer Protocol, on page 308
- Configuration Examples for SSH Support over IPv6, on page 309
- Additional References for SSH File Transfer Protocol, on page 310
- Feature History for SSH File Transfer Protocol, on page 310


## Prerequisites for SSH File Transfer Protocol

- SSH must be enabled.
- The **ip ssh source-interface** *interface-type interface-number* command must be configured.

## Restrictions for SSH File Transfer Protocol

- The SFTP server is not supported.
- SFTP boot is not supported.
- The **sftp** option in the **install add** command is not supported.

# Information About SSH Support over IPv6

## SSH File Transfer Protocol Overview

The SFTP client functionality is provided as part of the SSH component and is always enabled on the corresponding device. Therefore, any SFTP server user with the appropriate permission can copy files to and from the device.

An SFTP client is VRF-aware; you can configure the secure FTP client to use the virtual routing and forwarding (VRF) associated with a particular source interface during connection attempts.

# How to Configure SSH File Transfer Protocol

The following sections provide information about the various tasks that comprise an SFTP configuration.

## Configuring SFTP

Perform the following steps:

### Before you begin

To configure a Cisco device for SFTP client-side functionality, the **ip ssh source-interface** *interface-type interface-number* command must be configured first.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip ssh source-interface** *interface-type interface-number*<br>**Example:**<br>`Device(config)# ip ssh source-interface GigabitEthernet 1/0/1` | Defines the source IP for the SSH session. |
| **Step 4** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and returns to privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **show running-config**<br>**Example:**<br>`Device# show running-config` | (Optional) Displays the SFTP client-side functionality. |
| **Step 6** | **debug ip sftp**<br>**Example:**<br>`Device# debug ip sftp` | (Optional) Enables SFTP debugging. |

## Performing an SFTP Copy Operation

SFTP copy takes the IP or hostname of the corresponding server if Domain Name System (DNS) is configured. To perform SFTP copy operations, use the following commands in privileged EXEC mode:

| Command | Purpose |
|---|---|
| Device# **copy ios-file-system:file sftp://user:pwd@server-ip//filepath**<br>Or<br>Device# **copy ios-file-system: sftp:** | Copies a file from the local Cisco IOS file system to the server.<br>Specify the username, password, IP address, and filepath of the server. |
| Device# **copy sftp://user:pwd@server-ip //filepath ios-file-system:file**<br>Or<br>Device# **copy sftp: ios-file-system:** | Copies the file from the server to the local Cisco IOS file system.<br>Specify the username, password, IP address, and filepath of the server. |

# Configuration Examples for SSH Support over IPv6

## Example: Configuring SSH File Transfer Protocol

The following example shows how to configure the client-side functionality of SFTP:

```
Device> enable
Device# configure terminal
Device(config)# ip ssh source-interface gigabitethernet 1/0/1
Device(config)# exit
```

# Additional References for SSH File Transfer Protocol

### Related Documents

| Related Topic | Document Title |
|---|---|
| Secure Shell Version 1 and 2 Support | *Security Configuration Guide* |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/support |

# Feature History for SSH File Transfer Protocol

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Gibraltar 16.10.1 | SSH File Transfer Protocol | SSH includes support for SFTP, a new standard file transfer protocol introduced in SSHv2.<br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 18

# X.509v3 Certificates for SSH Authentication

• X.509v3 Certificates for SSH Authentication, on page 311

# X.509v3 Certificates for SSH Authentication

The X.509v3 Certificates for SSH Authentication feature uses the X.509v3 digital certificates in server and user authentication at the secure shell (SSH) server side.

This module describes how to configure server and user certificate profiles for a digital certificate.

## Prerequisites for X.509v3 Certificates for SSH Authentication

- The X.509v3 Certificates for SSH Authentication feature introduces the **ip ssh server algorithm authentication** command to replace the **ip ssh server authenticate user** command. If you use the **ip ssh server authenticate user** command, the following deprecation message is displayed.

  ```
  Warning: SSH command accepted but this CLI will be deprecated soon.
  Please move to new CLI "ip ssh server algorithm authentication".
  Please configure "default ip ssh server authenticate user" to make the CLI ineffective.
  ```

  Use the **default ip ssh server authenticate user** command to remove the **ip ssh server authenticate user** command from effect. The IOS secure shell (SSH) server then starts using the **ip ssh server algorithm authentication** command.

## Restrictions for X.509v3 Certificates for SSH Authentication

- The X.509v3 Certificates for SSH Authentication feature implementation is applicable only on the Cisco IOS XE secure shell (SSH) server side.
- The SSH server supports only the x509v3-ssh-rsa algorithm-based certificate for server and user authentication.

## Information About X.509v3 Certificates for SSH Authentication

The following section provides information about digital certificates, and server and user authentication.

## Digital Certificates

The validity of the authentication depends upon the strength of the linkage between the public signing key and the identity of the signer. Digital certificates in the X.509v3 format (RFC5280) are used to provide identity management. A chain of signatures by a trusted root certification authority and its intermediate certificate authorities binds a given public signing key to a given digital identity.

Public key infrastructure (PKI) trustpoint helps manage the digital certificates. The association between the certificate and the trustpoint helps track the certificate. The trustpoint contains information about the certificate authority (CA), different identity parameters, and the digital certificate. Multiple trustpoints can be created to associate with different certificates.

## Server and User Authentication using X.509v3

For server authentication, the Cisco IOS XE secure shell (SSH) server sends its own certificate to the SSH client for verification. This server certificate is associated with the trustpoint configured in the server certificate profile (ssh-server-cert-profile-server configuration mode).

For user authentication, the SSH client sends the user's certificate to the SSH server for verification. The SSH server validates the incoming user certificate using public key infrastructure (PKI) trustpoints configured in the server certificate profile (ssh-server-cert-profile-user configuration mode).

By default, certificate-based authentication is enabled for server and user at the SSH server end.

# How to Configure X.509v3 Certificates for SSH Authentication

The following section provides information about how to configure X.509v3 Certificates for SSH Authentication.

## Configuring the SSH Server to Use Digital Certificates for Server Authentication

To configure the SSH server to use digital certificates for server authentication, perform this procedure:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip ssh server algorithm hostkey** {**x509v3-ssh-rsa** [**ssh-rsa**] \| **ssh-rsa** [**x509v3-ssh-rsa**]}<br>**Example:**<br>`Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa` | Defines the order of host key algorithms. Only the configured algorithm is negotiated with the secure shell (SSH) client.<br>**Note**<br>The IOS SSH server must have at least one configured host key algorithm: |

| | Command or Action | Purpose |
|---|---|---|
| | | • ssh-rsa: public key based authentication<br>• x509v3-ssh-rsa: certificate-based authentication |
| **Step 4** | **ip ssh server certificate profile**<br>**Example:**<br>`Device(config)# ip ssh server certificate profile` | Configures server certificate profile and user certificate profile and enters SSH certificate profile configuration mode. |
| **Step 5** | **server**<br>**Example:**<br>`Device(ssh-server-cert-profile)# server` | Configures server certificate profile and enters SSH server certificate profile server configuration mode. |
| **Step 6** | **trustpoint sign** *PKI-trustpoint-name*<br>**Example:**<br>`Device(ssh-server-cert-profile-server)# trustpoint sign trust1` | Attaches the public key infrastructure (PKI) trustpoint to the server certificate profile. The SSH server uses the certificate associated with this PKI trustpoint for server authentication. |
| **Step 7** | **ocsp-response include**<br>**Example:**<br>`Device(ssh-server-cert-profile-server)# ocsp-response include` | (Optional) Sends the Online Certificate Status Protocol (OCSP) response or OCSP stapling along with the server certificate.<br>**Note**<br>By default the **no** form of this command is configured and no OCSP response is sent along with the server certificate. |
| **Step 8** | **end**<br>**Example:**<br>`Device(ssh-server-cert-profile-server)# end` | Exits SSH server certificate profile server configuration mode and returns to privileged EXEC mode. |

## Configuring the SSH Server to Verify Digital Certificates for User Authentication

To configure the SSH Server to use digital certificates for user authentication, perform this procedure:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | **ip ssh server algorithm authentication** {**publickey** \| **keyboard** \| **password**}<br>**Example:**<br>`Device(config)# ip ssh server algorithm authentication publickey` | Defines the order of user authentication algorithms. Only the configured algorithm is negotiated with the secure shell (SSH) client.<br>**Note**<br>• The SSH server must have at least one configured user authentication algorithm.<br>• To use the certificate method for user authentication, the **publickey** keyword must be configured.<br>• The **ip ssh server algorithm authentication** command replaces the **ip ssh server authenticate user** command. |
| **Step 4** | **ip ssh server algorithm publickey** {**x509v3-ssh-rsa** [**ssh-rsa**] \| **ssh-rsa** [**x509v3-ssh-rsa**]}<br>**Example:**<br>`Device(config)# ip ssh server algorithm publickey x509v3-ssh-rsa` | Defines the order of public key algorithms. Only the configured algorithm is accepted by the SSH client for user authentication.<br>**Note**<br>The SSH client must have at least one configured public key algorithm:<br>• ssh-rsa: public-key-based authentication<br>• x509v3-ssh-rsa: certificate-based authentication |
| **Step 5** | **ip ssh server certificate profile**<br>**Example:**<br>`Device(config)# ip ssh server certificate profile` | Configures server certificate profile and user certificate profile and enters SSH certificate profile configuration mode. |
| **Step 6** | **user**<br>**Example:**<br>`Device(ssh-server-cert-profile)# user` | Configures user certificate profile and enters SSH server certificate profile user configuration mode. |
| **Step 7** | **trustpoint verify** *PKI-trustpoint-name*<br>**Example:**<br>`Device(ssh-server-cert-profile-user)# trustpoint verify trust2` | Configures the public key infrastructure (PKI) trustpoint that is used to verify the incoming user certificate.<br>**Note**<br>Configure multiple trustpoints by executing the same command multiple times. A maximum of 10 trustpoints can be configured. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **ocsp-response required**<br>**Example:**<br>`Device(ssh-server-cert-profile-user)# ocsp-response required` | (Optional) Mandates the presence of the Online Certificate Status Protocol (OCSP) response with the incoming user certificate.<br>**Note**<br>By default the **no** form of this command is configured and the user certificate is accepted without an OCSP response. |
| **Step 9** | **end**<br>**Example:**<br>`Device(ssh-server-cert-profile-user)# end` | Exits SSH server certificate profile user configuration mode and returns to privileged EXEC mode. |

## Configuring Trustpoint Authentication and Creating Device Certificate

To configure trustpoint authentication and create device certificate, perform this procedure:

**Note** We recommend that you use a new RSA keypair name for the newly configured PKI certificate. If you want to reuse an existing RSA keypair name (that is associated with an old certificate) for a new PKI certificate, do either of the following:

- Do not regenerate a new RSA keypair with an existing RSA keypair name, reuse the existing RSA keypair name. Regenerating a new RSA keypair with an existing RSA keypair name will make all the certificates associated with the existing RSA keypair invalid.
- Manually remove the old PKI certificate configurations first, before reusing the existing RSA keypair name for the new PKI certificate.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **crypto pki trustpoint** *name*<br>**Example:**<br>`Device(config)# crypto pki trustpoint trust1` | Declares the trustpoint and a given name and enters ca-trustpoint configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **enrollment url** *url*<br>**Example:**<br>`Device(ca-trustpoint)# enrollment url http://10.1.1.10:80` | Specifies the URL of the CA on which your device should send certificate requests. |
| **Step 5** | **revocation-check none**<br>**Example:**<br>`Device(ca-trustpoint)# revocation-check none` | Specifies that certificate checking is ignored. |
| **Step 6** | **rsakeypair** *key-label* [*key-size* [*encryption-key-size*]]<br>**Example:**<br>`Device(ca-trustpoint)# rsakeypair trust1 2048` | (Optional) Specifies which key pair to associate with the certificate.<br>A key pair with the *key-label* argument will be generated during enrollment if it does not already exist or if the **auto-enroll regenerate** command was issued.<br>Specify the *key-size* argument for generating the key, and specify the *encryption-key-size* argument to request separate encryption, signature keys, and certificates. The *key-size* argument range is from 512 to 4096. The key-size and encryption-key-size must be the same size. Length of less than 2048 is not recommended.<br>**Note**<br>If this command is not enabled, the FQDN key pair is used. |
| **Step 7** | **exit**<br>**Example:**<br>`Device(ca-trustpoint)# exit` | Exits ca-trustpoint configuration mode and returns to global configuration mode. |
| **Step 8** | **crypto pki authenticate** *name*<br>**Example:**<br>`Device(config)# crypto pki authenticate trust1` | Retrieves the CA certificate and authenticates it. Check the certificate fingerprint if prompted.<br>**Note**<br>This command is optional if the CA certificate is already loaded into the configuration. |
| **Step 9** | **crypto pki enroll** *name*<br>**Example:**<br>`Device(config)# crypto pki enroll trust1` | Certificate request is sent to the certificate server and the server issues the ID or device certificate. You are prompted for enrollment information, such as whether to include the device FQDN and IP address in the certificate request. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 10** | **show crypto pki certificates**<br>**Example:**<br>`Device(config)# show crypto pki certificates verbose trust1` | (Optional) Displays information about your certificates, including any rollover certificates. |

### What to do next

For more information on how to install the certificate using other enrollment options, see Deploying RSA Keys Within a PKI.

# Verifying Configuration for Server and User Authentication Using Digital Certificates

To verify configuration for server and user Authentication using digital certificates, perform this procedure:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **show ip ssh**<br>**Example:**<br>`Device# show ip ssh`<br><br>`SSH Enabled - version 1.99`<br>`Authentication methods:publickey,keyboard-interactive,password`<br>`Authentication Publickey Algorithms:x509v3-ssh-rsa,ssh-rsa`<br>`Hostkey Algorithms:x509v3-ssh-rsa,ssh-rsa`<br>`Authentication timeout: 120 secs;`<br>`Authentication retries: 3`<br>`Minimum expected Diffie Hellman key size : 1024 bits` | Displays the currently configured authentication methods. To confirm the use of certificate-based authentication, ensure that the x509v3-ssh-rsa algorithm is the configured host key algorithm. |

# Configuration Examples for X.509v3 Certificates for SSH Authentication

The following section provides examples for user and server authentication using digital certificates.

## Example: Configuring the SSH Server to Use Digital Certificates for Server Authentication

This example shows how to configure the SSH Server to use digital certificates for server authentication.

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa
Device(config)# ip ssh server certificate profile
Device(ssh-server-cert-profile)# server
Device(ssh-server-cert-profile-server)# trustpoint sign trust1
Device(ssh-server-cert-profile-server)# end
```

## Example: Configuring the SSH Server to Verify Digital Certificates for User Authentication

This example shows how to configure the SSH server to verify user's digital certificate for user authentication.

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm authentication publickey
Device(config)# ip ssh server algorithm publickey x509v3-ssh-rsa
Device(config)# ip ssh server certificate profile
Device(ssh-server-cert-profile)# user
Device(ssh-server-cert-profile-user)# trustpoint verify trust2
Device(ssh-server-cert-profile-user)# end
```

# Feature History for X.509v3 Certificates for SSH Authentication

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | X.509v3 Certificates for SSH Authentication | The X.509v3 Certificates for SSH Authentication feature uses the X.509v3 digital certificates in server and user authentication at the SSH server side.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | X.509v3 Certificates for SSH Authentication | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 19

# SSH Algorithms for Common Criteria Certification


- Restriction for SSH Algorithms for Common Criteria Certification, on page 319
- Information About SSH Algorithms for Common Criteria Certification, on page 319
- How to Configure SSH Algorithms for Common Criteria Certification, on page 321
- Configuration Examples For SSH Algorithms for Common Criteria Certification, on page 325
- Verifying SSH Algorithms for Common Criteria Certification , on page 326
- Feature History for Secure Shell Algorithms for Common Criteria Certification , on page 327


# Restriction for SSH Algorithms for Common Criteria Certification

Starting from Cisco IOS XE Amsterdam 17.1.1, SHA1 is not supported.

# Information About SSH Algorithms for Common Criteria Certification

This section provides information about the Secure Shell (SSH) Algorithms for Common Criteria Certification, the Cisco IOS SSH Server Algorithms and Cisco IOS SSH Client Algorithms.

## SSH Algorithms for Common Criteria Certification

A Secure Shell (SSH) configuration enables a Cisco IOS SSH server and client to authorize the negotiation of only those algorithms that are configured from the allowed list. If a remote party tries to negotiate using only those algorithms that are not part of the allowed list, the request is rejected and the session is not established.

## Cisco IOS SSH Server Algorithms

Cisco IOS secure shell (SSH) servers support the encryption algorithms (Advanced Encryption Standard Counter Mode [AES-CTR], AES Cipher Block Chaining [AES-CBC], Triple Data Encryption Standard [3DES]) in the following order:

Supported Default Encryption Order:

1. aes128-gcm

2. aes256-gcm
3. aes128-ctr
4. aes192-ctr
5. aes256-ctr

Supported Non-Default Encrytion Order:

1. aes128-cbc
2. aes192-cbc
3. aes256-cbc
4. 3des

Cisco IOS SSH clients support the Message Authentication Code (MAC) algorithms in the following order:

Supported Default HMAC order:

1. hmac-sha2-256
2. hmac-sha2-512

Cisco IOS SSH clients support only one host key algorithm and do not need a CLI configuration.

Supported Default Host Key order:

1. x509v3-ssh-rsa
2. ssh-rsa

# Cisco IOS SSH Client Algorithms

Cisco IOS secure shell (SSH) clients support the encryption algorithms (Advanced Encryption Standard counter mode [AES-CTR], AES Cipher Block Chaining [AES-CBC], Triple Data Encryption Standard [3DES]) in the following order:

Supported Default Encryption Order:

1. aes128-gcm
2. aes256-gcm
3. aes128-ctr
4. aes192-ctr
5. aes256-ctr

Supported Non-Default Encrytion Order:

1. aes128-cbc
2. aes192-cbc
3. aes256-cbc

4. 3des

Cisco IOS SSH clients support the Message Authentication Code (MAC) algorithms in the following order:

Supported Default HMAC order:

1. hmac-sha2-256
2. hmac-sha2-512

Cisco IOS SSH clients support only one host key algorithm and do not need a CLI configuration.

Supported Default Host Key order:

1. x509v3-ssh-rsa
2. ssh-rsa

# How to Configure SSH Algorithms for Common Criteria Certification

This section provides information on how to configure and troubleshoot:

- Encryption key algorithm for a Cisco IOS SSH server and client
- MAC algorithm for a Cisco IOS SSH server and client
- Host Key algorithm for a Cisco IOS SSH server

## Configuring an Encryption Key Algorithm for a Cisco IOS SSH Server and Client

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip ssh** {**server** \| **client**} **algorithm encryption** {**aes128-gcm** \| **aes256-gcm** \| **aes128-ctr** \| **aes192-ctr** \| **aes256-ctr** \| **aes128cbc** \| **aes192-cbs** \| **3des**}<br>**Example:** | Defines the order of encryption algorithms in the SSH server and client. This order is presented during algorithm negotiation.<br>**Note** |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# ip ssh server algorithm encryption aes128-gcm aes256-gcm aes128-ctr aes192-ctr aes256-ctr aes128-cbc aes192-cbc aes256-cbc 3des`<br>`Device(config)# ip ssh client algorithm encryption aes128-gcm aes256-gcm aes128-ctr aes192-ctr aes256-ctr aes128-cbc aes192-cbc aes256-cbc 3des` | The Cisco IOS SSH server and client must have at least one configured encryption algorithm.<br>**Note**<br>To disable one algorithm from the previously configured algorithm list, use the **no** form of this command. To disable more than one algorithm, use the **no** form of this command multiple times with different algorithm names.<br>**Note**<br>For a default configuration, use the default form of this command as shown below:<br>`Device(config)# ip ssh server algorithm encryption aes128-gcm aes256-gcm aes128-ctr aes192-ctr aes256-ctr aes128-cbc aes192-cbc aes256-cbc 3des` |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

## Troubleshooting Tips

If you try to disable the last encryption algorithm in the configuration, the following message is displayed and the command is rejected:

```
% SSH command rejected: All encryption algorithms cannot be disabled
```

# Configuring a MAC Algorithm for a Cisco IOS SSH Server and Client

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | **ip ssh** {**server** \| **client**} **algorithm mac** {**hmac-sha2-256-etm** \| **hmac-sha2-512-etm** \| **hmac-sha2-256** \| **hmac-sha2-512** }<br>**Example:**<br>`Device(config)# ip ssh server algorithm mac hmac-sha2-256-etm hmac-sha2-512-etm hmac-sha2-256 hmac-sha2-512`<br>`Device(config)# ip ssh client algorithm mac hmac-sha2-256-etm hmac-sha2-512-etm hmac-sha2-256 hmac-sha2-512` | Defines the order of MAC (Message Authentication Code) algorithms in the SSH server and client. This order is presented during algorithm negotiation.<br>**Note**<br>The Cisco IOS SSH server and client must have at least one configured Hashed Message Authentication Code (HMAC) algorithm.<br>**Note**<br>To disable one algorithm from the previously configured algorithm list, use the **no** form of this command. To disable more than one algorithm, use the **no** form of this command multiple times with different algorithm names.<br>**Note**<br>For default configuration, use the default form of this command as shown below:<br>`Device(config)# ip ssh server algorithm mac hmac-sha2-256-etm hmac-sha2-512-etm hmac-sha2-256 hmac-sha2-512` |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

## Troubleshooting Tips

If you try to disable the last MAC algorithm in the configuration, the following message is displayed and the command is rejected:

```
% SSH command rejected: All mac algorithms cannot be disabled
```

# Configuring a Host Key Algorithm for a Cisco IOS SSH Server

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** | Enables privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br><br>`Device> enable` | Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br><br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip ssh server algorithm hostkey** {**x509v3-ssh-rsa** \|**rsa-sha2-512**\| **rsa-sha2-256ssh-rsa**}<br>**Example:**<br><br>`Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa rsa-sha2-512 rsa-sha2-256 ssh-rsa` | Defines the order of host key algorithms. Only the configured algorithm is negotiated with the Cisco IOS secure shell (SSH) client.<br>**Note**<br>The Cisco IOS SSH server must have at least one configured host key algorithm:<br>• x509v3-ssh-rsa—X.509v3 certificate-based authentication<br>• ssh-rsa—Public-key-based authentication<br><br>**Note**<br>To disable one algorithm from the previously configured algorithm list, use the **no** form of this command. To disable more than one algorithm, use the **no** form of this command multiple times with different algorithm names.<br><br>**Note**<br>For default configuration, use the default form of this command as shown below:<br><br>`Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa rsa-sha2-512 rsa-sha2-256 ssh-rsa` |
| **Step 4** | **end**<br>**Example:**<br><br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

## Troubleshooting Tips

If you try to disable the last host key algorithm in the configuration, the following message is displayed and the command is rejected:

```
% SSH command rejected: All hostkey algorithms cannot be disabled
```

# Configuration Examples For SSH Algorithms for Common Criteria Certification

This section provides configuration examples for SSH algorithms for common certification.

## Example: Configuring Encryption Key Algorithms for a Cisco IOS SSH Server

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm encryption aes128-ctr aes192-ctr aes256-ctr aes128-cbc
 aes192-cbc aes256-cbc 3des
Device(config)# end
```

## Example: Configuring Encryption Key Algorithms for a Cisco IOS SSH Client

```
Device> enable
Device# configure terminal
Device(config)# ip ssh client algorithm encryption aes128-ctr aes192-ctr aes256-ctr aes128-cbc
 aes192-cbc aes256-cbc 3des

Device(config)# end
```

## Example: Configuring MAC Algorithms for a Cisco IOS SSH Server

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm mac hmac-sha2-256-etm hmac-sha2-512-etm hmac-sha2-256
 hmac-sha2-512
Device(config)# end
```

## Example: Configuring Host Key Algorithms for a Cisco IOS SSH Server

```
Device> enable
Device# configure terminal
Device(config)# ip ssh server algorithm hostkey x509v3-ssh-rsa rsa-sha2-512 rsa-sha2-256
ssh-rsaa
Device(config)# end
```

# Verifying SSH Algorithms for Common Criteria Certification

## Procedure

**Step 1** **enable**

Enables privileged EXEC mode.

- Enter your password if prompted.

**Example:**

```
Device> enable
```

**Step 2** **show ip ssh**

Displays configured Secure Shell (SSH) encryption, host key, and Message Authentication Code (MAC) algorithms.

**Example:**

The following sample output from the **show ip ssh** command shows the encryption algorithms configured in the default order:

```
Device# show ip ssh

Encryption Algorithms: aes128-ctr aes192-ctr aes256-ctr aes128-cbc aes192-cbc aes256-cbc
3des
```

The following sample output from the **show ip ssh** command shows the MAC algorithms configured in the default order:

```
Device# show ip ssh

MAC Algorithms: hmac-sha2-256, hmac-sha2-512
```

The following sample output from the **show ip ssh** command shows the host key algorithms configured in the default order:

```
Device# show ip ssh

Hostkey Algorithms: x509v3-ssh-rsa, ssh-rsa
```

# Feature History for Secure Shell Algorithms for Common Criteria Certification

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Secure Shell Algorithms for Common Criteria Certification | The SSH Algorithms for Common Criteria Certification feature provides the list and order of the algorithms that are allowed for Common Criteria Certification. This module describes how to configure the encryption, Message Authentication Code (MAC), and host key algorithms for a secure shell (SSH) server and client so that SSH connections can be limited on the basis of the allowed algorithms list.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Gibraltar 16.11.1 | Secure Shell Algorithms for Common Criteria Certification | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 20

# Configuring Secure Socket Layer HTTP


- Information About Secure Socket Layer HTTP, on page 329
- How to Configure Secure Socket Layer HTTP, on page 332
- Monitoring Secure HTTP Server and Client Status, on page 338
- Additional References for Secure Socket Layer HTTP, on page 339
- Feature History for Secure Socket Layer HTTP, on page 339


# Information About Secure Socket Layer HTTP

## Secure HTTP Servers and Clients Overview

On a secure HTTP connection, data to and from an HTTP server is encrypted before being sent over the Internet. HTTP with SSL encryption provides a secure connection to allow such functions as configuring a switch from a Web browser. Cisco's implementation of the secure HTTP server and secure HTTP client uses an implementation of SSL Version 3.0 with application-layer encryption. HTTP over SSL is abbreviated as HTTPS; the URL of a secure connection begins with https:// instead of http://.

The primary role of the HTTP secure server (the switch) is to listen for HTTPS requests on a designated port (the default HTTPS port is 443) and pass the request to the HTTP 1.1 Web server. The HTTP 1.1 server processes requests and passes responses (pages) back to the HTTP secure server, which, in turn, responds to the original request.

The primary role of the HTTP secure client (the web browser) is to respond to Cisco IOS application requests for HTTPS User Agent services, perform HTTPS User Agent services for the application, and pass the response back to the application.

## Certificate Authority Trustpoints

Certificate authorities (CAs) manage certificate requests and issue certificates to participating network devices. These services provide centralized security key and certificate management for the participating devices. Specific CA servers are referred to as *trustpoints*.

When a connection attempt is made, the HTTPS server provides a secure connection by issuing a certified X.509v3 certificate, obtained from a specified CA trustpoint, to the client. The client (usually a Web browser), in turn, has a public key that allows it to authenticate the certificate.

For secure HTTP connections, we highly recommend that you configure a CA trustpoint. If a CA trustpoint is not configured for the device running the HTTPS server, the server certifies itself and generates the needed RSA key pair. Because a self-certified (self-signed) certificate does not provide adequate security, the connecting client generates a notification that the certificate is self-certified, and the user has the opportunity to accept or reject the connection. This option is useful for internal network topologies (such as testing).

If you do not configure a CA trustpoint, when you enable a secure HTTP connection, either a temporary or a persistent self-signed certificate for the secure HTTP server (or client) is automatically generated.

- If the switch is not configured with a hostname and a domain name, a temporary self-signed certificate is generated. If the switch reboots, any temporary self-signed certificate is lost, and a new temporary new self-signed certificate is assigned.
- If the switch has been configured with a host and domain name, a persistent self-signed certificate is generated. This certificate remains active if you reboot the switch or if you disable the secure HTTP server so that it will be there the next time you re-enable a secure HTTP connection.

**Note** The certificate authorities and trustpoints must be configured on each device individually. Copying them from other devices makes them invalid on the switch.

When a new certificate is enrolled, the new configuration change is not applied to the HTTPS server until the server is restarted. You can restart the server using the **reload** command. On restarting the server, the switch starts using the new certificate.

If a self-signed certificate has been generated, this information is included in the output of the **show running-config** privileged EXEC command. This is a partial sample output from that command displaying a self-signed certificate.

```
Device# show running-config

Building configuration...

<output truncated>

crypto pki trustpoint TP-self-signed-3080755072
 enrollment selfsigned
 subject-name cn=IOS-Self-Signed-Certificate-3080755072
 revocation-check none
 rsakeypair TP-self-signed-3080755072
!
!
crypto ca certificate chain TP-self-signed-3080755072
 certificate self-signed 01
  3082029F 30820208 A0030201 02020101 300D0609 2A864886 F70D0101 04050030
  59312F30 2D060355 04031326 494F532D 53656C66 2D536967 6E65642D 43657274
  69666963 6174652D 33303830 37353530 37323126 30240609 2A864886 F70D0109
  02161743 45322D33 3535302D 31332E73 756D6D30 342D3335 3530301E 170D3933
  30333031 30303030 35395A17 0D323030 31303130 30303030 305A3059 312F302D

<output truncated>
```

You can remove this self-signed certificate by disabling the secure HTTP server and entering the **no crypto pki trustpoint TP-self-signed-30890755072** global configuration command. If you later re-enable a secure HTTP server, a new self-signed certificate is generated.

**Note** The values that follow *TP self-signed* depend on the serial number of the device.

You can use the **ip http secure-client-auth** command (optional) to allow the HTTPS server to request an X.509v3 certificate from the client. Authenticating the client provides more security than server authentication by itself.

**Note** Beginning from Cisco IOS XE Amsterdam 17.3.x, CA's self-signed root certificate must be configured for successful authentication of the client.

# CipherSuites

A CipherSuite specifies the encryption algorithm and the digest algorithm to use on a SSL connection. When connecting to the HTTPS server, the client Web browser offers a list of supported CipherSuites, and the client and server negotiate the best encryption algorithm to use from those on the list that are supported by both. For example, Netscape Communicator 4.76 supports U.S. security with RSA Public Key Cryptography, MD2, MD5, RC2-CBC, RC4, DES-CBC, and DES-EDE3-CBC.

For the best possible encryption, you should use a client browser that supports 128-bit encryption, such as Microsoft Internet Explorer Version 5.5 (or later) or Netscape Communicator Version 4.76 (or later). The SSL_RSA_WITH_DES_CBC_SHA CipherSuite provides less security than the other CipherSuites, as it does not offer 128-bit encryption.

The more secure and more complex CipherSuites require slightly more processing time. This list defines the CipherSuites supported by the switch and ranks them from fastest to slowest in terms of router processing load (speed):

1. SSL_RSA_WITH_DES_CBC_SHA—RSA key exchange (RSA Public Key Cryptography) with DES-CBC for message encryption and SHA for message digest
2. SSL_RSA_WITH_NULL_SHA key exchange with NULL for message encryption and SHA for message digest (only for SSL 3.0).
3. SSL_RSA_WITH_NULL_MD5 key exchange with NULL for message encryption and MD5 for message digest (only for SSL 3.0).
4. SSL_RSA_WITH_RC4_128_MD5—RSA key exchange with RC4 128-bit encryption and MD5 for message digest
5. SSL_RSA_WITH_RC4_128_SHA—RSA key exchange with RC4 128-bit encryption and SHA for message digest
6. SSL_RSA_WITH_3DES_EDE_CBC_SHA—RSA key exchange with 3DES and DES-EDE3-CBC for message encryption and SHA for message digest
7. SSL_RSA_WITH_AES_128_CBC_SHA—RSA key exchange with AES 128-bit encryption and SHA for message digest (only for SSL 3.0).
8. SSL_RSA_WITH_AES_256_CBC_SHA—RSA key exchange with AES 256-bit encryption and SHA for message digest (only for SSL 3.0).

9. SSL_RSA_WITH_DHE_AES_128_CBC_SHA—RSA key exchange with AES 128-bit encryption and SHA for message digest (only for SSL 3.0).
10. SSL_RSA_WITH_DHE_AES_256_CBC_SHA—RSA key exchange with AES 256-bit encryption and SHA for message digest (only for SSL 3.0).

**Note** The latest versions of Chrome do not support the four original cipher suites, thus disallowing access to both web GUI and guest portals.

RSA (in conjunction with the specified encryption and digest algorithm combinations) is used for both key generation and authentication on SSL connections. This usage is independent of whether or not a CA trustpoint is configured.

## Default SSL Configuration

The following guidelines apply to the default SSL configuration:

- The standard HTTP server is enabled.
- SSL is enabled.
- No CA trustpoints are configured.
- No self-signed certificates are generated.

## SSL Configuration Guidelines

When SSL is used in a switch cluster, the SSL session terminates at the cluster commander. Cluster member switches must run standard HTTP.

Before you configure a CA trustpoint, you should ensure that the system clock is set. If the clock is not set, the certificate is rejected due to an incorrect date.

In a switch stack, the SSL session terminates at the active switch.

# How to Configure Secure Socket Layer HTTP

## Configuring a CA Trustpoint

For secure HTTP connections, we recommend that you configure an official CA trustpoint. A CA trustpoint is more secure than a self-signed certificate.

Beginning in privileged EXEC mode, follow these steps to configure a CA Trustpoint:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **hostname** *hostname*<br>**Example:**<br>`Device(config)# hostname your_hostname` | Specifies the hostname of the switch (required only if you have not previously configured a hostname). The hostname is required for security keys and certificates. |
| **Step 4** | **ip domain name** *domain-name*<br>**Example:**<br>`Device(config)# ip domain name your_domain` | Specifies the IP domain name of the switch (required only if you have not previously configured an IP domain name). The domain name is required for security keys and certificates. |
| **Step 5** | **crypto key generate rsa**<br>**Example:**<br>`Device(config)# crypto key generate rsa` | (Optional) Generates an RSA key pair. RSA key pairs are required before you can obtain a certificate for the switch. RSA key pairs are generated automatically. You can use this command to regenerate the keys, if needed. |
| **Step 6** | **crypto ca trustpoint** *name*<br>**Example:**<br>`Device(config)# crypto ca trustpoint your_trustpoint` | Specifies a local configuration name for the CA trustpoint and enter CA trustpoint configuration mode. |
| **Step 7** | **enrollment url** *url*<br>**Example:**<br>`Device(ca-trustpoint)# enrollment url http://your_server:80` | Specifies the URL to which the switch should send certificate requests. |
| **Step 8** | **enrollment http-proxy** *host-name port-number*<br>**Example:**<br>`Device(ca-trustpoint)# enrollment http-proxy your_host 49` | (Optional) Configures the switch to obtain certificates from the CA through an HTTP proxy server.<br>• For *host-name* , specify the proxy server used to get the CA.<br>• For *port-number*, specify the port number used to access the CA. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **crl query** *url*<br>**Example:**<br>`Device(ca-trustpoint)# crl query ldap://your_host:49` | Configures the switch to request a certificate revocation list (CRL) to ensure that the certificate of the peer has not been revoked. |
| **Step 10** | **primary** *name*<br>**Example:**<br>`Device(ca-trustpoint)# primary your_trustpoint` | (Optional) Specifies that the trustpoint should be used as the primary (default) trustpoint for CA requests.<br>• For *name*, specify the trustpoint that you just configured. |
| **Step 11** | **exit**<br>**Example:**<br>`Device(ca-trustpoint)# exit` | Exits CA trustpoint configuration mode and return to global configuration mode. |
| **Step 12** | **crypto ca authentication** *name*<br>**Example:**<br>`Device(config)# crypto ca authentication your_trustpoint` | Authenticates the CA by getting the public key of the CA. Use the same name used in Step 5. |
| **Step 13** | **crypto ca enroll** *name*<br>**Example:**<br>`Device(config)# crypto ca enroll your_trustpoint` | Obtains the certificate from the specified CA trustpoint. This command requests a signed certificate for each RSA key pair. |
| **Step 14** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring the Secure HTTP Server

Beginning in privileged EXEC mode, follow these steps to configure a secure HTTP server:

### Before you begin

If you are using a certificate authority for certification, you should use the previous procedure to configure the CA trustpoint on the switch before enabling the HTTP server. If you have not configured a CA trustpoint, a self-signed certificate is generated the first time that you enable the secure HTTP server. After you have configured the server, you can configure options (path, access list to apply, maximum number of connections, or timeout policy) that apply to both standard and secure HTTP servers.

To verify the secure HTTP connection by using a Web browser, enter https://*URL*, where the URL is the IP address or hostname of the server switch. If you configure a port other than the default port, you must also specify the port number after the URL. For example:

**Note** AES256_SHA2 is not supported.

```
https://209.165.129:1026
```

or

```
https://host.domain.com:1026
```

The existing **ip http access-class** *access-list-number* command for specifying the access-list(Only IPv4 ACLs) is going to be deprecated. You can still use this command to specify an access list to allow access to the HTTP server. Two new commands have been introduced to enable support for specifying IPv4 and IPv6 ACLs. These are **ip http access-class ipv4** *access-list-name* | *access-list-number* for specifying IPv4 ACLs and **ip http access-class ipv6** *access-list-name* for specifying IPv6 ACLs. We recommend using the new CLI to avoid receiving warning messages.

Note the following considerations for specifying access-lists:

- If you specify an access-list that does not exist, the configuration takes place but you receive the below warning message:

  ```
  ACL being attached does not exist, please configure it
  ```

- If you use **ip http access-class ipv4** *access-list-name* | *access-list-number* or **ip http access-class ipv6** *access-list-name* , and an access-list was already configured using **ip http access-class** , the below warning message appears:

  ```
  Removing ip http access-class <access-list-number>
  ```

**ip http access-class** *access-list-number* and **ip http access-class ipv4** *access-list-name* | *access-list-number* share the same functionality. Each command overrides the configuration of the previous command. The following combinations between the configuration of the two commands explain the effect on the running configuration:

- If **ip http access-class** *access-list-number* is already configured and you try to configure using **ip http access-class ipv4** *access-list-number* command, the configuration of **ip http access-class** *access-list-number* will be removed and the configuration of **ip http access-class ipv4** *access-list-number* will be added to the running configuration.

- If **ip http access-class** *access-list-number* is already configured and you try to configure using **ip http access-class ipv4** *access-list-name* command, the configuration of **ip http access-class** *access-list-number* will be removed and the configuration of **ip http access-class ipv4** *access-list-name* will be added to the running configuration.

- If **ip http access-class ipv4** *access-list-number* is already configured and you try to configure using **ip http access-class** *access-list-name*, the configuration of **ip http access-class ipv4** *access-list-number* will be removed from configuration and the configuration of **ip http access-class** *access-list-name* will be added to the running configuration.

- If **ip http access-class ipv4** *access-list-name* is already configured and you try to configure using **ip http access-class** *access-list-number*, the configuration of **ip http access-class ipv4** *access-list-name* will be removed from the configuration and the configuration of **ip http access-class** *access-list-number* will be added to the running configuration.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **show ip http server status**<br>**Example:**<br>`Device# show ip http server status` | (Optional) Displays the status of the HTTP server to determine if the secure HTTP server feature is supported in the software. You should see one of these lines in the output:<br>`HTTP secure server capability: Present`<br>or<br>`HTTP secure server capability: Not present` |
| **Step 3** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 4** | **ip http secure-server**<br>**Example:**<br>`Device(config)# ip http secure-server` | Enables the HTTPS server if it has been disabled. The HTTPS server is enabled by default. |
| **Step 5** | **ip http secure-port** *port-number*<br>**Example:**<br>`Device(config)# ip http secure-port 443` | (Optional) Specifies the port number to be used for the HTTPS server. The default port number is 443. Valid options are 443 or any number in the range 1025 to 65535. |
| **Step 6** | **ip http secure-ciphersuite** {[**3des-ede-cbc-sha**] [**rc4-128-md5**] [**rc4-128-sha**] [**des-cbc-sha**]}<br>**Example:**<br>`Device(config)# ip http secure-ciphersuite rc4-128-md5` | (Optional) Specifies the CipherSuites (encryption algorithms) to be used for encryption over the HTTPS connection. If you do not have a reason to specify a particularly CipherSuite, you should allow the server and client to negotiate a CipherSuite that they both support. This is the default. |
| **Step 7** | **ip http secure-client-auth**<br>**Example:**<br>`Device(config)# ip http secure-client-auth` | (Optional) Configures the HTTP server to request an X.509v3 certificate from the client for authentication during the connection process. The default is for the client to request a certificate from the server, but the server does not attempt to authenticate the client. |
| **Step 8** | **ip http secure-trustpoint** *name*<br>**Example:**<br>`Device(config)# ip http secure-trustpoint your_trustpoint` | Specifies the CA trustpoint to use to get an X.509v3 security certificate and to authenticate the client certificate connection.<br>**Note** |

| | Command or Action | Purpose |
|---|---|---|
| | | Use of this command assumes you have already configured a CA trustpoint according to the previous procedure. |
| **Step 9** | **ip http path** *path-name*<br>**Example:**<br>`Device(config)# ip http path /your_server:80` | (Optional) Sets a base HTTP path for HTML files. The path specifies the location of the HTTP server files on the local system (usually located in system flash memory). |
| **Step 10** | **ip http access-class** { **ipv4** {*access-list-number* \| *access-list-name*} \| **ipv6** {*access-list-name*} }<br>**Example:**<br>`Device(config)# ip http access-class ipv4 4` | (Optional)Specifies an access list to use to allow access to the HTTP server. |
| **Step 11** | **ip http max-connections** *value*<br>**Example:**<br>`Device(config)# ip http max-connections 4` | (Optional) Sets the maximum number of concurrent connections that are allowed to the HTTP server. We recommend that the value be at least 10 and not less. This is required for the UI to function as expected. |
| **Step 12** | **ip http timeout-policy idle** *seconds* **life** *seconds* **requests** *value*<br>**Example:**<br>`Device(config)# ip http timeout-policy idle 120 life 240 requests 1` | (Optional) Specifies how long a connection to the HTTP server can remain open under the defined circumstances:<br>• **idle**: the maximum time period when no data is received or response data cannot be sent. The range is 1 to 600 seconds. The default is 180 seconds (3 minutes).<br>• **life**: the maximum time period from the time that the connection is established. The range is 1 to 86400 seconds (24 hours). The default is 180 seconds.<br>• **requests**: the maximum number of requests processed on a persistent connection. The maximum value is 86400. The default is 1. |
| **Step 13** | **end**<br>**Example:**<br>`Device(config)#end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring the Secure HTTP Client

Beginning in privileged EXEC mode, follow these steps to configure a secure HTTP client:

### Before you begin

The standard HTTP client and secure HTTP client are always enabled. A certificate authority is required for secure HTTP client certification. This procedure assumes that you have previously configured a CA trustpoint on the switch. If a CA trustpoint is not configured and the remote HTTPS server requires client authentication, connections to the secure HTTP client fail.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip http client secure-trustpoint** *name*<br>**Example:**<br>`Device(config)# ip http client secure-trustpoint your_trustpoint` | (Optional) Specifies the CA trustpoint to be used if the remote HTTP server requests client authentication. Using this command assumes that you have already configured a CA trustpoint by using the previous procedure. The command is optional if client authentication is not needed or if a primary trustpoint has been configured. |
| **Step 4** | **ip http client secure-ciphersuite** {[**3des-ede-cbc-sha**] [**rc4-128-md5**] [**rc4-128-sha**] [**des-cbc-sha**]}<br>**Example:**<br>`Device(config)# ip http client secure-ciphersuite rc4-128-md5` | (Optional) Specifies the CipherSuites (encryption algorithms) to be used for encryption over the HTTPS connection. If you do not have a reason to specify a particular CipherSuite, you should allow the server and client to negotiate a CipherSuite that they both support. This is the default. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)#end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Monitoring Secure HTTP Server and Client Status

To monitor the SSL secure server and client status, use the privileged EXEC commands in the following table.

*Table 20: Commands for Displaying the SSL Secure Server and Client Status*

| Command | Purpose |
|---|---|
| **show ip http client secure status** | Shows the HTTP secure client configuration. |

| Command | Purpose |
|---|---|
| **show ip http server secure status** | Shows the HTTP secure server configuration. |
| **show running-config** | Shows the generated self-signed certificate for secure HTTP connections. |

# Additional References for Secure Socket Layer HTTP

### Related Documents

| Related Topic | Document Title |
|---|---|
| Certification Authority | Configuring Certification Authority Interoperability |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/support |

# Feature History for Secure Socket Layer HTTP

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Secure Socket Layer HTTP | Cisco's implementation of the secure HTTP server and secure HTTP client uses an implementation of SSL Version 3.0 with application-layer encryption. On a secure HTTP connection, data to and from an HTTP server is encrypted before being sent over the Internet.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Secure Socket Layer HTTP | Support for this feature was introduced on the C9500-24Y4C, C9500-32C, C9500-32QC, and C9500-48Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 21

# IPv4 ACLs


- Restrictions for IPv4 Access Control Lists, on page 341
- Information About IPv4 Access Control Lists, on page 342
- How to Configure IPv4 Access Control Lists, on page 353
- Monitoring IPv4 ACLs, on page 369
- Configuration Examples for IPv4 Access Control Lists, on page 369
- Additional References for IPv4 Access Control Lists, on page 381
- Feature History for IPv4 Access Control Lists, on page 381


## Restrictions for IPv4 Access Control Lists

### General Network Security

The following are restrictions for configuring network security with ACLs:

- Not all commands that accept a numbered ACL accept a named ACL. ACLs for packet filters and route filters on interfaces can use a name. VLAN maps also accept a name.
- A standard ACL and an extended ACL cannot have the same name.
- Though visible in the command-line help strings, **appletalk** is not supported as a matching condition for the **deny** and **permit** MAC access-list configuration mode commands.
- ACL wildcard is not supported in downstream client policy.
- ACLs cannot be configured on management ports.
- When you apply a scale ACL to an interface that does not program TCAM for a protocol and the ACLs that have been unloaded, it can impact the existing normal movement of traffic for other protocols. The restriction is applicable to IPv6 and MAC address traffic for C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series SwitchesIPv4, IPv6, and MAC address traffic for C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches.
- Router ACL is enforced on all types of traffic, including CPU generated traffic.
- ACL logging in the egress direction are not supported for packets that are generated from the control plane of the device.
- Time-to-live (TTL) classification is not supported on ACLs.

- If a downloadable ACL contains any type of duplicate entries, the entries are not auto merged. As a result, the 802.1X session authorization fails. Ensure that the downloadable ACL is optimized without any duplicate entries, for example port-based and name-based entries for the same port.
- Egress ACL lookup is not supported for injected traffic that is forwarded by the software.

### IPv4 ACL Network Interfaces

The following restrictions apply to IPv4 ACLs to network interfaces:

- When controlling access to an interface, you can use a named or numbered ACL.
- If you apply an ACL to a Layer 2 interface that is a member of a VLAN, the Layer 2 (port) ACL takes precedence over an input Layer 3 ACL applied to the VLAN interface or a VLAN map applied to the VLAN.
- If you apply an ACL to a Layer 3 interface and routing is not enabled on the switch, the ACL only filters packets that are intended for the CPU, such as SNMP, Telnet, or web traffic.
- If the **preauth_ipv4_acl** ACL is configured to filter packets, the ACL is removed after authentication.
- You do not have to enable routing to apply ACLs to Layer 2 interfaces.

### MAC ACLs on a Layer 2 Interface

After you create a MAC ACL, you can apply it to a Layer 2 interface to filter non-IP traffic coming in that interface. When you apply the MAC ACL, consider these guidelines:

- You can apply no more than one IP access list and one MAC access list to the same Layer 2 interface. The IP access list filters only IP packets, and the MAC access list filters non-IP packets.
- A Layer 2 interface can have only one MAC access list. If you apply a MAC access list to a Layer 2 interface that has a MAC ACL configured, the new ACL replaces the previously configured one.

**Note** The **mac access-group** interface configuration command is only valid when applied to a physical Layer 2 interface. You cannot use the command on EtherChannel port channels.

### IP Access List Entry Sequence Numbering

- This feature does not support dynamic, reflexive, or firewall access lists.

# Information About IPv4 Access Control Lists

## ACL Overview

Packet filtering can help limit network traffic and restrict network use by certain users or devices. ACLs filter traffic as it passes through a device and permit or deny packets crossing specified interfaces. An ACL is a sequential collection of permit and deny conditions that apply to packets. When a packet is received on an

interface, the switch compares the fields in the packet against any applied ACLs to verify that the packet has the required permissions to be forwarded, based on the criteria specified in the access lists. One by one, it tests packets against the conditions in an access list. The first match decides whether the switch accepts or rejects the packets. Because the switch stops testing after the first match, the order of conditions in the list is critical. If no conditions match, the switch rejects the packet. If there are no restrictions, the switch forwards the packet; otherwise, the switch drops the packet. The switch can use ACLs on all packets it forwards.

You configure access lists on a device to provide basic security for your network. If you do not configure ACLs, all packets passing through the switch could be allowed onto all parts of the network. You can use ACLs to control which hosts can access different parts of a network or to decide which types of traffic are forwarded or blocked at device interfaces. For example, you can allow e-mail traffic to be forwarded but not Telnet traffic.

# Access Control Entries

An ACL contains an ordered list of access control entries (ACEs). Each ACE specifies *permit* or *deny* and a set of conditions the packet must satisfy in order to match the ACE. The meaning of *permit* or *deny* depends on the context in which the ACL is used.

# ACL Supported Types

The device supports IP ACLs and Ethernet (MAC) ACLs:

- IP ACLs filter IPv4 traffic, including TCP, User Datagram Protocol (UDP), Internet Group Management Protocol (IGMP), and Internet Control Message Protocol (ICMP).
- Ethernet ACLs filter non-IP traffic.

This device also supports quality of service (QoS) classification ACLs.

# Supported ACLs

The switch supports three types of ACLs to filter the traffic:

- Port ACLs access-control traffic entering a Layer 2 interface. You can apply port ACLs to a Layer 2 interface in each direction to each access list type—IPv4 and MAC.
- Router ACLs access-control traffic routed between VLANs and are applied to Layer 3 interfaces in a specific direction (inbound or outbound).
- VLAN ACLs or VLAN maps are applied only to Layer 2 VLANs and impact bridged traffic only. You can use VLAN maps to filter traffic between devices in the same VLAN. VLAN maps are configured to provide access control based on Layer 3 addresses for IPv4. Unsupported protocols are access-controlled through MAC addresses using Ethernet ACEs. After a VLAN map is applied to a VLAN, all packets (routed or bridged) entering the VLAN are checked against the VLAN map. Packets can either enter the VLAN through a switch port or through a routed port after being routed.

## ACL Precedence

When VLAN maps, Port ACLs, and router ACLs are configured on the same switch, the filtering precedence, from greatest to least for ingress traffic is port ACL, VLAN map, and then router ACL. For egress traffic, the filtering precedence is router ACL, VLAN map, and then port ACL.

The following examples describe simple use cases:

- When both an input port ACL and a VLAN map are applied, incoming packets received on ports with a port ACL applied are filtered by the port ACL. Other packets are filtered by the VLAN map
- When an input router ACL and input port ACL exist in a switch virtual interface (SVI), incoming packets received on ports to which a port ACL is applied are filtered by the port ACL. Incoming routed IP packets received on other ports are filtered by the router ACL. Other packets are not filtered.
- When an output router ACL and input port ACL exist in an SVI, incoming packets received on the ports to which a port ACL is applied are filtered by the port ACL. Outgoing routed IP packets are filtered by the router ACL. Other packets are not filtered.
- When a VLAN map, input router ACL, and input port ACL exist in an SVI, incoming packets received on the ports to which a port ACL is applied are only filtered by the port ACL. Incoming routed IP packets received on other ports are filtered by both the VLAN map and the router ACL. Other packets are filtered only by the VLAN map.
- When a VLAN map, output router ACL, and input port ACL exist in an SVI, incoming packets received on the ports to which a port ACL is applied are only filtered by the port ACL. Outgoing routed IP packets are filtered by both the VLAN map and the router ACL. Other packets are filtered only by the VLAN map.

## Port ACLs

Port ACLs are ACLs that are applied to Layer 2 interfaces on a switch. Port ACLs are supported on physical interfaces and EtherChannel interfaces but not on EtherChannel member interfaces. Port ACLs can be applied to the interface in inbound and outbound direction. The following access lists are supported:

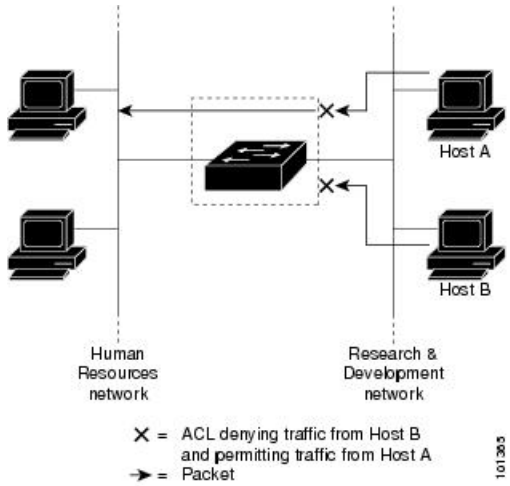
- Standard IP access lists using source addresses
- Extended IP access lists using source and destination addresses and optional protocol type information
- MAC extended access lists using source and destination MAC addresses and optional protocol type information

The switch examines ACLs on an interface and permits or denies packet forwarding based on how the packet matches the entries in the ACL. In this way, ACLs control access to a network or to part of a network.

***Figure 15: Using ACLs to Control Traffic in a Network***

This is an example of using port ACLs to control access to a network when all workstations are in the same VLAN. ACLs applied at the Layer 2 input would allow Host A to access the Human Resources network, but

prevent Host B from accessing the same network. Port ACLs can only be applied to Layer 2 interfaces in the




inbound direction.

When you apply a port ACL to a trunk port, the ACL filters traffic on all VLANs present on the trunk port. When you apply a port ACL to a port with voice VLAN, the ACL filters traffic on both data and voice VLANs.

With port ACLs, you can filter IP traffic by using IP access lists and non-IP traffic by using MAC addresses. You can filter both IP and non-IP traffic on the same Layer 2 interface by applying both an IP access list and a MAC access list to the interface.

**Note** You can't apply more than one IP access list and one MAC access list to a Layer 2 interface. If an IP access list or MAC access list is already configured on a Layer 2 interface and you apply a new IP access list or MAC access list to the interface, the new ACL replaces the previously configured one.

## Router ACLs

You can apply router ACLs on switch virtual interfaces (SVIs), which are Layer 3 interfaces to VLANs; on physical Layer 3 interfaces; and on Layer 3 EtherChannel interfaces. You apply router ACLs on interfaces for specific directions (inbound or outbound). You can apply one router ACL in each direction on an interface.

The switch supports these access lists for IPv4 traffic:

- Standard IP access lists use source addresses for matching operations.
- Extended IP access lists use source and destination addresses and optional protocol type information for matching operations.

As with port ACLs, the switch examines ACLs associated with features configured on a given interface. As packets enter the switch on an interface, ACLs associated with all inbound features configured on that interface are examined. After packets are routed and before they are forwarded to the next hop, all ACLs associated with outbound features configured on the egress interface are examined.

ACLs permit or deny packet forwarding based on how the packet matches the entries in the ACL, and can be used to control access to a network or to part of a network.

## VLAN Maps

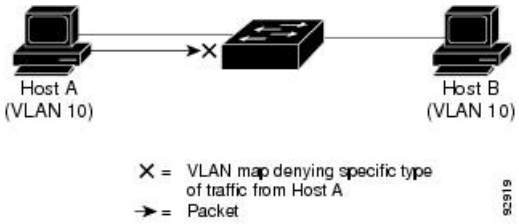
VLAN ACLs or VLAN maps are used to control the network traffic within a VLAN. You can apply VLAN maps to all packets that are bridged within a VLAN in the switch or switch stack. VACLs are strictly for the security packet filtering and for redirecting traffic to specific physical interfaces. VACLs are not defined by direction (ingress or egress).

All non-IP protocols are access-controlled through MAC addresses and Ethertype using MAC VLAN maps. (IP traffic is not access-controlled by MAC VLAN maps.) You can enforce VLAN maps only on packets going through the switch; you cannot enforce VLAN maps on traffic between hosts on a hub or on another switch that is connected to this switch.

With VLAN maps, forwarding of packets is permitted or denied, based on the action specified in the map.

***Figure 16: Using VLAN Maps to Control Traffic***

This figure shows how a VLAN map is applied to prevent a specific type of traffic from Host A in VLAN 10 from being forwarded. You can apply only one VLAN map to a VLAN.




# ACEs and Fragmented and Unfragmented Traffic

IP packets can be fragmented as they cross the network. When this happens, only the fragment containing the beginning of the packet contains the Layer 4 information, such as TCP or UDP port numbers, ICMP type and code, and so on. All other fragments are missing this information.

Some access control entries (ACEs) do not check Layer 4 information and therefore can be applied to all packet fragments. ACEs that do test Layer 4 information cannot be applied in the standard manner to most of the fragments in a fragmented IP packet. When the fragment contains no Layer 4 information and the ACE tests some Layer 4 information, the matching rules are modified:

- Permit ACEs that check the Layer 3 information in the fragment (including protocol type, such as TCP, UDP, and so on) are considered to match the fragment regardless of what the missing Layer 4 information might have been.

**Note** For TCP ACEs with L4 Ops, the fragmented packets will be dropped per RFC 1858.

- Deny ACEs that check Layer 4 information never match a fragment unless the fragment contains Layer 4 information.

# Standard and Extended IPv4 ACLs

An ACL is a sequential collection of permit and deny conditions. One by one, the device tests packets against the conditions in an access list. The first match determines whether the device accepts or rejects the packet. Because the device stops testing after the first match, the order of the conditions is critical. If no conditions match, the device denies the packet.

The software supports these types of ACLs or access lists for IPv4:

- Standard IP access lists use source addresses for matching operations.
- Extended IP access lists use source and destination addresses for matching operations and optional protocol-type information for finer granularity of control.

## IPv4 ACL Switch Unsupported Features

The following ACL-related features are not supported:

- Non-IP protocol ACLs
- IP accounting
- Reflexive ACLs and dynamic ACLs are not supported.

## Access List Numbers

The number you use to denote your ACL shows the type of access list that you are creating.

This lists the access-list number and corresponding access list type and shows whether or not they are supported in the switch. The switch supports IPv4 standard and extended access lists, numbers 1 to 199 and 1300 to 2699.

***Table 21: Access List Numbers***

| Access List Number | Type | Supported |
|---|---|---|
| 1–99 | IP standard access list | Yes |
| 100–199 | IP extended access list | Yes |
| 200–299 | Protocol type-code access list | No |
| 300–399 | DECnet access list | No |
| 400–499 | XNS standard access list | No |
| 500–599 | XNS extended access list | No |
| 600–699 | AppleTalk access list | No |
| 700–799 | 48-bit MAC address access list | No |
| 800–899 | IPX standard access list | No |
| 900–999 | IPX extended access list | No |

| Access List Number | Type | Supported |
|---|---|---|
| 1000–1099 | IPX SAP access list | No |
| 1100–1199 | Extended 48-bit MAC address access list | No |
| 1200–1299 | IPX summary address access list | No |
| 1300–1999 | IP standard access list (expanded range) | Yes |
| 2000–2699 | IP extended access list (expanded range) | Yes |

In addition to numbered standard and extended ACLs, you can also create standard and extended named IP ACLs by using the supported numbers. That is, the name of a standard IP ACL can be 1 to 99; the name of an extended IP ACL can be 100 to 199. The advantage of using named ACLs instead of numbered lists is that you can delete individual entries from a named list.

## Numbered Standard IPv4 ACLs

When creating an ACL, remember that, by default, the end of the ACL contains an implicit deny statement for all packets that it did not find a match for before reaching the end. With standard access lists, if you omit the mask from an associated IP host address ACL specification, 0.0.0.0 is assumed to be the mask.

Starting from the Cisco IOS XE Bengaluru 17.5.1 release, when using the **show ip access-list** *acl_name* or the **show run section** *acl_name* command, the ACEs are displayed in ascending order according to their sequence numbers.

After creating a numbered standard IPv4 ACL, you can apply it to VLANs, to terminal lines, or to interfaces.

## Numbered Extended IPv4 ACLs

Although standard ACLs use only source addresses for matching, you can use extended ACL source and destination addresses for matching operations and optional protocol type information for finer granularity of control. When you are creating ACEs in numbered extended access lists, remember that after you create the ACL, any additions are placed at the end of the list. You cannot reorder the list or selectively add or remove ACEs from a numbered list.

The device does not support dynamic or reflexive access lists. It also does not support filtering based on the type of service (ToS) minimize-monetary-cost bit.

Some protocols also have specific parameters and keywords that apply to that protocol.

You can define an extended TCP, UDP, ICMP, IGMP, or other IP ACL. The device also supports these IP protocols:

**Note** ICMP echo-reply cannot be filtered. All other ICMP codes or types can be filtered.

These IP protocols are supported:

- Authentication Header Protocol (**ahp**)
- Encapsulation Security Payload (**esp**)
- Enhanced Interior Gateway Routing Protocol (**eigrp**)

- Generic routing encapsulation (**gre**)
- Internet Control Message Protocol (**icmp**)
- Internet Group Management Protocol (**igmp**)
- Any Interior Protocol (**ip**)
- IP in IP tunneling (**ipinip**)
- KA9Q NOS-compatible IP over IP tunneling (**nos**)
- Open Shortest Path First routing (**ospf**)
- Payload Compression Protocol (**pcp**)
- Protocol-Independent Multicast (**pim**)
- Transmission Control Protocol (**tcp**)
- User Datagram Protocol (**udp**)

## Named IPv4 ACLs

You can identify IPv4 ACLs with an alphanumeric string (a name) rather than a number. You can use named ACLs to configure more IPv4 access lists in a device than if you were to use numbered access lists. If you identify your access list with a name rather than a number, the mode and command syntax are slightly different. However, not all commands that use IP access lists accept a named access list.

**Note** The name you give to a standard or extended ACL can also be a number in the supported range of access list numbers. That is, the name of a standard IP ACL can be 1 to 99. The advantage of using named ACLs instead of numbered lists is that you can delete individual entries from a named list.

Consider these guidelines before configuring named ACLs:

- Numbered ACLs are also available.
- A standard ACL and an extended ACL cannot have the same name.
- You can use standard or extended ACLs (named or numbered) in VLAN maps.

## ACL Logging

The device software can provide logging messages about packets permitted or denied by a standard IP access list. That is, any packet that matches the ACL causes an informational logging message about the packet to be sent to the console. The level of messages logged to the console is controlled by the **logging console** commands controlling the syslog messages.

**Note** ACL logging is not supported for ACLs used with Unicast Reverse Path Forwarding (uRPF). It is only supported for router ACL.

**Note** Because routing is done in hardware and logging is done in software, if a large number of packets match a *permit* or *deny* ACE containing a **log** keyword, the software might not be able to match the hardware processing rate, and not all packets will be logged.

The first packet that triggers the ACL causes a logging message right away, and subsequent packets are collected over 5-minute intervals before they appear or logged. The logging message includes the access list number, whether the packet was permitted or denied, the source IP address of the packet, and the number of packets from that source permitted or denied in the prior 5-minute interval.

**Note** The logging facility might drop some logging message packets if there are too many to be handled or if there is more than one logging message to be handled in 1 second. This behavior prevents the device from crashing due to too many logging packets. Therefore, the logging facility should not be used as a billing tool or an accurate source of the number of matches to an access list.

# Hardware and Software Treatment of IP ACLs

ACL processing is performed in hardware. If the hardware reaches its capacity to store ACL configurations, all packets on that interface are dropped.

**Note** If an ACL configuration cannot be implemented in hardware due to an out-of-resource condition on a device or stack member, then only the traffic in that VLAN arriving on that device is affected.

For router ACLs, other factors can cause packets to be sent to the CPU:

- Using the **log** keyword
- Generating ICMP unreachable messages

When you enter the **show ip access-lists** privileged EXEC command, the match count displayed does not account for packets that are access controlled in hardware. Use the **show platform software fed switch** { *switch_num* | **active** | **standby** } **acl counters hardware** privileged EXEC command to obtain some basic hardware ACL statistics for switched and routed packets.

Router ACLs function as follows:

- The hardware controls permit and deny actions of standard and extended ACLs (input and output) for security access control.
- If **log** has not been specified, the flows that match a *deny* statement in a security ACL are dropped by the hardware if *ip unreachables* is disabled. The flows matching a *permit* statement are switched in hardware.
- Adding the **log** keyword to an ACE in a router ACL causes a copy of the packet to be sent to the CPU for logging only. If the ACE is a *permit* statement, the packet is still switched and routed in hardware.

# VLAN Map Configuration Guidelines

VLAN maps are the only way to control filtering within a VLAN. VLAN maps have no direction. To filter traffic in a specific direction by using a VLAN map, you need to include an ACL with specific source or destination addresses. If there is a match clause for that type of packet (IP or MAC) in the VLAN map, the default action is to drop the packet if the packet does not match any of the entries within the map. If there is no match clause for that type of packet, the default is to forward the packet.

The following are the VLAN map configuration guidelines:

- If there is no ACL configured to deny traffic on an interface and no VLAN map is configured, all traffic is permitted.
- Each VLAN map consists of a series of entries. The order of entries in an VLAN map is important. A packet that comes into the device is tested against the first entry in the VLAN map. If it matches, the action specified for that part of the VLAN map is taken. If there is no match, the packet is tested against the next entry in the map.
- If the VLAN map has at least one match clause for the type of packet (IP or MAC) and the packet does not match any of these match clauses, the default is to drop the packet. If there is no match clause for that type of packet in the VLAN map, the default is to forward the packet.
- Logging is not supported for VLAN maps.
- When a device has an IP access list or MAC access list applied to a Layer 2 interface, and you apply a VLAN map to a VLAN that the port belongs to, the port ACL takes precedence over the VLAN map.
- If a VLAN map configuration cannot be applied in hardware, all packets in that VLAN are dropped.

# VLAN Maps with Router ACLs

To access control both bridged and routed traffic, you can use VLAN maps only or a combination of router ACLs and VLAN maps. You can define router ACLs on both input and output routed VLAN interfaces, and you can define a VLAN map to access control the bridged traffic.

If a packet flow matches a VLAN-map deny clause in the ACL, regardless of the router ACL configuration, the packet flow is denied.

**Note** When you use router ACLs with VLAN maps, packets that require logging on the router ACLs are not logged if they are denied by a VLAN map.

If the VLAN map has a match clause for the type of packet (IP or MAC) and the packet does not match the type, the default is to drop the packet. If there is no match clause in the VLAN map, and no action specified, the packet is forwarded if it does not match any VLAN map entry.

## VLAN Maps and Router ACL Configuration Guidelines

These guidelines are for configurations where you need to have an router ACL and a VLAN map on the same VLAN. These guidelines do not apply to configurations where you are mapping router ACLs and VLAN maps on different VLANs.

If you must configure a router ACL and a VLAN map on the same VLAN, use these guidelines for both router ACL and VLAN map configuration:

- You can configure only one VLAN map and one router ACL in each direction (input/output) on a VLAN interface.
- Whenever possible, try to write the ACL with all entries having a single action except for the final, default action of the other type. That is, write the ACL using one of these two forms:

  permit... permit... permit... deny ip any any

  or

  deny... deny... deny... permit ip any any
- To define multiple actions in an ACL (permit, deny), group each action type together to reduce the number of entries.
- Avoid including Layer 4 information in an ACL; adding this information complicates the merging process. The best merge results are obtained if the ACLs are filtered based on IP addresses (source and destination) and not on the full flow (source IP address, destination IP address, protocol, and protocol ports). It is also helpful to use *don't care* bits in the IP address, whenever possible.

  If you need to specify the full-flow mode and the ACL contains both IP ACEs and TCP/UDP/ICMP ACEs with Layer 4 information, put the Layer 4 ACEs at the end of the list. This gives priority to the filtering of traffic based on IP addresses.

# Time Ranges for ACLs

You can selectively apply extended ACLs based on the time of day and the week by using the **time-range** global configuration command. First, define a time-range name and set the times and the dates or the days of the week in the time range. Then enter the time-range name when applying an ACL to set restrictions to the access list. You can use the time range to define when the permit or deny statements in the ACL are in effect, for example, during a specified time period or on specified days of the week. The **time-range** keyword and argument are referenced in the named and numbered extended ACL task tables.

These are some benefits of using time ranges:

- You have more control over permitting or denying a user access to resources, such as an application (identified by an IP address/mask pair and a port number).
- You can control logging messages. ACL entries can be set to log traffic only at certain times of the day. Therefore, you can simply deny access without needing to analyze many logs generated during peak hours.

Time-based access lists trigger CPU activity because the new configuration of the access list must be merged with other features and the combined configuration loaded into the hardware memory. For this reason, you should be careful not to have several access lists configured to take affect in close succession (within a small number of minutes of each other.)

**Note** The time range relies on the device system clock; therefore, you need a reliable clock source. We recommend that you use Network Time Protocol (NTP) to synchronize the device clock.

# IPv4 ACL Interface Considerations

When you apply the **ip access-group** interface configuration command to a Layer 3 interface (an SVI, a Layer 3 EtherChannel, or a routed port), the interface must have been configured with an IP address. Layer 3 access groups filter packets that are routed or are received by Layer 3 processes on the CPU. They do not affect packets bridged within a VLAN.

For inbound ACLs, after receiving a packet, the device checks the packet against the ACL. If the ACL permits the packet, the device continues to process the packet. If the ACL rejects the packet, the device discards the packet.

For outbound ACLs, after receiving and routing a packet to a controlled interface, the device checks the packet against the ACL. If the ACL permits the packet, the device sends the packet. If the ACL rejects the packet, the device discards the packet.

By default, the input interface sends ICMP Unreachable messages whenever a packet is discarded, regardless of whether the packet was discarded because of an ACL on the input interface or because of an ACL on the output interface. ICMP Unreachables are normally limited to no more than one every one-half second per input interface, but this can be changed by using the **ip icmp rate-limit unreachable** global configuration command.

When you apply an undefined ACL to an interface, the device acts as if the ACL has not been applied to the interface and permits all packets. Remember this behavior if you use undefined ACLs for network security.

# How to Configure IPv4 Access Control Lists

## Configuring IPv4 ACLs

These are the steps to use IP ACLs on the switch:

### Procedure

**Step 1** Create an ACL by specifying an access list number or name and the access conditions.

**Step 2** Apply the ACL to interfaces or terminal lines. You can also apply standard and extended IP ACLs to VLAN maps.

## Creating a Numbered Standard ACL

Follow these steps to create a numbered standard ACL:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** | Enables privileged EXEC mode. |
| | **Example:** | Enter your password, if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device> enable` | |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **access-list** *access-list-number* {**deny** \| **permit**} *source source-wildcard* ]<br>**Example:**<br>`Device(config)# access-list 2 deny your_host` | Defines a standard IPv4 access list by using a source address and wildcard.<br>The *access-list-number* is a decimal number from 1 to 99 or 1300 to 1999.<br>Enter **deny** or **permit** to specify whether to deny or permit access if conditions are matched.<br>The *source* is the source address of the network or host from which the packet is being sent specified as:<br>• The 32-bit quantity in dotted-decimal format.<br>• The keyword **any** as an abbreviation for *source* and *source-wildcard* of 0.0.0.0 255.255.255.255. You do not need to enter a source-wildcard.<br>• The keyword **host** as an abbreviation for source and *source-wildcard* of *source* 0.0.0.0.<br>(Optional) The *source-wildcard* applies wildcard bits to the source.<br>**Note**<br>Logging is supported only on ACLs attached to Layer 3 interfaces. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Creating a Numbered Extended ACL

Follow these steps to create a numbered extended ACL:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** | Enables privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device> enable` | Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **access-list** *access-list-number* {**deny** \| **permit**} *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**tos** tos] [**fragments**] [**time-range** *time-range-name*] [**dscp** *dscp*]<br>**Example:**<br>`Device(config)# access-list 101 permit ip host 10.1.1.2 any precedence 0 tos 0 log` | Defines an extended IPv4 access list and the access conditions.<br>The *access-list-number* is a decimal number from 100 to 199 or 2000 to 2699.<br>Enter **deny** or **permit** to specify whether to deny or permit the packet if conditions are matched.<br>Source, source-wildcard, destination, and destination-wildcard can be specified as:<br>• The 32-bit quantity in dotted-decimal format.<br>• The keyword **any** for 0.0.0.0 255.255.255.255 (any host).<br>• The keyword **host** for a single host 0.0.0.0.<br>The other keywords are optional and have these meanings:<br>• **precedence**: Enter to match packets with a precedence level specified as a number from 0 to 7 or by name: **routine** (0), **priority** (1), **immediate** (2), **flash** (3), **flash-override** (4), **critical** (5), **internet** (6), **network** (7).<br>• **fragments**: Enter to check non-initial fragments.<br>• **tos**: Enter to match by type of service level, specified by a number from 0 to 15 or a name: **normal** (0), **max-reliability** (2), **max-throughput** (4), **min-delay** (8).<br>• **time-range**: Specify the time-range name.<br>• **dscp**: Enter to match packets with the DSCP value specified by a number from 0 to 63, or use the question mark (?) to see a list of available values.<br>**Note** |

| | Command or Action | Purpose |
|---|---|---|
| | | If you enter a **dscp** value, you cannot enter **tos** or **precedence**. You can enter both a **tos** and a **precedence** value with no **dscp**. |
| **Step 4** | **access-list** *access-list-number* {**deny** \| **permit**} **tcp** *source source-wildcard* [*operator port*] *destination destination-wildcard* [*operator port*] [**established**] [**precedence** *precedence*] [**tos** *tos*] [**fragments**] [**time-range** *time-range-name*] [**dscp** *dscp*] [*flag*]<br>**Example:**<br>`Device(config)# access-list 101 permit tcp any any eq 500` | Defines an extended TCP access list and the access conditions.<br>The parameters are the same as those described for an extended IPv4 ACL, with these exceptions:<br>(Optional) Enter an *operator* and *port* to compare source (if positioned after *source source-wildcard*) or destination (if positioned after *destination destination-wildcard*) port. Possible operators include **eq** (equal), **gt** (greater than), **lt** (less than), **neq** (not equal), and **range** (inclusive range). Operators require a port number (range requires two port numbers separated by a space).<br>Enter the *port* number as a decimal number (from 0 to 65535) or the name of a TCP port. Use only TCP port numbers or names when filtering TCP.<br>The other optional keywords have these meanings:<br>• **established**: Enter to match an established connection. This has the same function as matching on the **ack** or **rst** flag.<br>• *flag*: Enter one of these flags to match by the specified TCP header bits: **ack** (acknowledge), **fin** (finish), **psh** (push), **rst** (reset), **syn** (synchronize), or **urg** (urgent). |
| **Step 5** | **access-list** *access-list-number* {**deny** \| **permit**} **udp** *source source-wildcard* [*operator port*] *destination destination-wildcard* [*operator port*] [**precedence** *precedence*] [**tos** *tos*] [**fragments**] [**time-range** *time-range-name*] [**dscp** *dscp*]<br>**Example:**<br>`Device(config)# access-list 101 permit udp any any eq 100` | (Optional) Defines an extended UDP access list and the access conditions.<br>The UDP parameters are the same as those described for TCP except that the [operator [port]] port number or name must be a UDP port number or name, and the **flag** and **established** keywords are not valid for UDP. |
| **Step 6** | **access-list** *access-list-number* {**deny** \| **permit**} **icmp** *source source-wildcard destination destination-wildcard* [*icmp-type* \| [[*icmp-type* | Defines an extended ICMP access list and the access conditions. |

| | Command or Action | Purpose |
|---|---|---|
| | *icmp-code*] \| [*icmp-message*]] [**precedence** *precedence*] [**tos** *tos*] [**fragments**] [**time-range** *time-range-name*] [**dscp** *dscp*]<br>**Example:**<br>`Device(config)# access-list 101 permit icmp any any 200` | The ICMP parameters are the same as those described for most IP protocols in an extended IPv4 ACL, with the addition of the ICMP message type and code parameters. These optional keywords have these meanings:<br>• *icmp-type*: Enter to filter by ICMP message type, a number from 0 to 255.<br>• *icmp-code*: Enter to filter ICMP packets that are filtered by the ICMP message code type, a number from 0 to 255.<br>• *icmp-message*: Enter to filter ICMP packets by the ICMP message type name or the ICMP message type and code name. |
| **Step 7** | **access-list** *access-list-number* {**deny** \| **permit**} **igmp** *source source-wildcard destination destination-wildcard* [*igmp-type*] [**precedence** *precedence*] [**tos** *tos*] [**fragments**] [**time-range** *time-range-name*] [**dscp** *dscp*]<br>**Example:**<br>`Device(config)# access-list 101 permit igmp any any 14` | (Optional) Defines an extended IGMP access list and the access conditions.<br>The IGMP parameters are the same as those described for most IP protocols in an extended IPv4 ACL, with this optional parameter.<br>*igmp-type*: To match IGMP message type, enter a number from 0 to 15, or enter the message name: **dvmrp**, **host-query**, **host-report**, **pim**, or **trace**. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Creating Named Standard ACLs

Follow these steps to create a standard ACL using names:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **ip access-list standard** *name*<br>**Example:**<br>`Device(config)# ip access-list standard 20` | Defines a standard IPv4 access list using a name, and enter access-list configuration mode.<br>The name can be a number from 1 to 99. |
| **Step 4** | Use one of the following:<br>• **deny** {*source* [*source-wildcard*] \| **host** *source* \| **any**} [**log**]<br>• **permit** {*source* [*source-wildcard*] \| **host** *source* \| **any**} [**log**]<br>**Example:**<br>`Device(config-std-nacl)# deny 192.168.0.0 0.0.255.255 255.255.0.0 0.0.255.255`<br>or<br>`Device(config-std-nacl)# permit 10.108.0.0 0.0.0.0 255.255.255.0 0.0.0.0` | In access-list configuration mode, specify one or more conditions denied or permitted to decide if the packet is forwarded or dropped.<br>• **host** *source*: A source and source wildcard of *source* 0.0.0.0.<br>• **any**: A source and source wildcard of 0.0.0.0 255.255.255.255. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-std-nacl)# end` | Exits access-list configuration mode and returns to privileged EXEC mode. |

# Creating Extended Named ACLs

Follow these steps to create an extended ACL using names:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip access-list extended** *name*<br>**Example:**<br>`Device(config)# ip access-list extended 150` | Defines an extended IPv4 access list using a name, and enter access-list configuration mode.<br>The name can be a number from 100 to 199. |
| **Step 4** | {**deny** \| **permit**} *protocol* {*source* [*source-wildcard*] \| **host** *source* \| **any**} | In access-list configuration mode, specify the conditions allowed or denied. Use the **log** |

| | Command or Action | Purpose |
|---|---|---|
| | {*destination* [*destination-wildcard*] \| host *destination* \| **any**} [**precedence** *precedence*] [**tos** *tos*] [**established**] [**log**] [**time-range** *time-range-name*]<br>**Example:**<br>`Device(config-ext-nacl)# permit 0 any any` | keyword to get access list logging messages, including violations.<br>• **host** *source*: A source and source wildcard of *source* 0.0.0.0.<br>• **host** *destintation*: A destination and destination wildcard of *destination* 0.0.0.0.<br>• **any**: A source and source wildcard or destination and destination wildcard of 0.0.0.0 255.255.255.255. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-ext-nacl)# end` | Exits access-list configuration mode and returns to privileged EXEC mode. |

When you are creating extended ACLs, remember that, by default, the end of the ACL contains an implicit deny statement for everything if it did not find a match before reaching the end. For standard ACLs, if you omit the mask from an associated IP host address access list specification, 0.0.0.0 is assumed to be the mask.

After you create an ACL, any additions are placed at the end of the list. You cannot selectively add ACL entries to a specific ACL. However, you can use **no permit** and **no deny** access-list configuration mode commands to remove entries from a named ACL.

Being able to selectively remove lines from a named ACL is one reason you might use named ACLs instead of numbered ACLs.

### What to do next

After creating a named ACL, you can apply it to interfaces or to VLANs .

# Configuring Time Ranges for ACLs

Follow these steps to configure a time-range parameter for an ACL:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **time-range** *time-range-name*<br>**Example:**<br>`Device(config)# time-range workhours` | Assigns a meaningful name (for example, *workhours*) to the time range to be created, and enters time-range configuration mode. The name cannot contain a space or quotation mark and must begin with a letter. |
| **Step 4** | Use one of the following:<br>• **absolute** [**start** *time date*] [**end** *time date*]<br>• **periodic** *day-of-the-week hh:mm to* [*day-of-the-week*] *hh:mm*<br>• **periodic** {**weekdays** \| **weekend** \| **daily**} *hh:mm to hh:mm*<br>**Example:**<br>`Device(config-time-range)# absolute start 00:00 1 Jan 2006 end 23:59 1 Jan 2006`<br>or<br>`Device(config-time-range)# periodic weekdays 8:00 to 12:00` | Specifies when the function it will be applied to is operational.<br>• You can use only one **absolute** statement in the time range. If you configure more than one absolute statement, only the one configured last is executed.<br>• You can enter multiple **periodic** statements. For example, you could configure different hours for weekdays and weekends. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-time-range)# end` | Exits time-range configuration mode and returns to privileged EXEC mode. |

### What to do next

Repeat the steps if you have multiple items that you want in effect at different times.

# Applying an IPv4 ACL to a Terminal Line

You can use numbered ACLs to control access to one or more terminal lines. You cannot apply named ACLs to lines. You must set identical restrictions on all the virtual terminal lines because a user can attempt to connect to any of them.

Follow these steps to restrict incoming and outgoing connections between a virtual terminal line and the addresses in an ACL:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | **line** [**console** \| **vty**] *line-number*<br>**Example:**<br>`Device(config)# line console 0` | Identifies a specific line to configure, and enter in-line configuration mode.<br>• **console**: Specifies the console terminal line. The console port is DCE.<br>• **vty**: Specifies a virtual terminal for remote console access.<br>The *line-number* is the first line number in a contiguous group that you want to configure when the line type is specified. The range is from 0 to 16. |
| **Step 4** | **access-class** *access-list-number* {**in** \| **out**}<br>**Example:**<br>`Device(config-line)# access-class 10 in` | Restricts incoming and outgoing connections between a particular virtual terminal line (into a device) and the addresses in an access list. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-line)# end` | Exits line configuration mode and returns to privileged EXEC mode. |

# Applying an IPv4 ACL to an Interface

This section describes how to apply IPv4 ACLs to network interfaces.

Beginning in privileged EXEC mode, follow these steps to control access to an interface:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet1/0/1` | Identifies a specific interface for configuration, and enters interface configuration mode.<br>The interface can be a Layer 2 interface (port ACL), or a Layer 3 interface (router ACL). |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **ip access-group** {*access-list-number* \| *name*} {**in** \| **out**}<br>**Example:**<br>`Device(config-if)# ip access-group 2 in` | Controls access to the specified interface. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Creating Named MAC Extended ACLs

You can filter non-IPv4 traffic on a VLAN or on a Layer 2 interface by using MAC addresses and named MAC extended ACLs. The procedure is similar to that of configuring other extended named ACLs.

Follow these steps to create a named MAC extended ACL:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **mac access-list extended** *name*<br>**Example:**<br>`Device(config)# mac access-list extended mac1` | Defines an extended MAC access list using a name. |
| **Step 4** | {**deny** \| **permit**} {**any** \| **host** *source MAC address* \| *source MAC address mask*} {**any** \| **host** *destination MAC address* \| *destination MAC address mask*} [*type mask* \| **lsap** *lsap mask* \| **aarp** \| **amber** \| **dec-spanning** \| **decnet-iv** \| **diagnostic** \| **dsm** \| **etype-6000** \| **etype-8042** \| **lat** \| **lavc-sca** \| **mop-console** \| **mop-dump** \| | In extended MAC access-list configuration mode, specifies to **permit** or **deny** any source MAC address, a source MAC address with a mask, or a specific **host** source MAC address and **any** destination MAC address, destination MAC address with a mask, or a specific destination MAC address. |

| | Command or Action | Purpose |
|---|---|---|
| | **msdos** \| **mumps** \| **netbios** \| **vines-echo** \| **vines-ip** \| **xns-idp** \| 0-*65535*] [**cos** *cos*]<br>**Example:**<br>`Device(config-ext-macl)# deny any any decnet-iv`<br>or<br>`Device(config-ext-macl)# permit any any` | (Optional) You can also enter these options:<br>• *type mask*—An arbitrary EtherType number of a packet with Ethernet II or SNAP encapsulation in decimal, hexadecimal, or octal with optional mask of *don't care* bits applied to the EtherType before testing for a match.<br>• **lsap** *lsap mask*—An LSAP number of a packet with IEEE 802.2 encapsulation in decimal, hexadecimal, or octal with optional mask of *don't care* bits.<br>• **aarp** \| **amber** \| **dec-spanning** \| **decnet-iv** \| **diagnostic** \| **dsm** \| **etype-6000** \| **etype-8042** \| **lat** \| **lavc-sca** \| **mop-console** \| **mop-dump** \| **msdos** \| **mumps** \| **netbios** \| **vines-echo** \| **vines-ip** \| **xns-idp**—A non-IP protocol.<br>• **cos** *cos*—An IEEE 802.1Q cost of service number from 0 to 7 used to set priority. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-ext-macl)# end` | Exits extended MAC access-list configuration mode and returns to privileged EXEC mode. |

# Applying a MAC ACL to a Layer 2 Interface

Follow these steps to apply a MAC access list to control access to a Layer 2 interface:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet1/0/2` | Identifies a specific interface, and enters interface configuration mode. The interface must be a physical Layer 2 interface (port ACL). |
| **Step 4** | **mac access-group** {*name*} {**in** \| **out** }<br>**Example:**<br>`Device(config-if)# mac access-group mac1 in` | Controls access to the specified interface by using the MAC access list.<br>Port ACLs are supported in the outbound and inbound directions . |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show mac access-group** [**interface** *interface-id*]<br>**Example:**<br>`Device# show mac access-group interface gigabitethernet1/0/2` | Displays the MAC access list applied to the interface or all Layer 2 interfaces. |

After receiving a packet, the device checks it against the inbound ACL. If the ACL permits it, the device continues to process the packet. If the ACL rejects the packet, the device discards it. When you apply an undefined ACL to an interface, the device acts as if the ACL has not been applied and permits all packets. Remember this behavior if you use undefined ACLs for network security.

# Configuring an IPv4 ACL in Template Mode

**Note** You can configure **ip access-group** command in the template configuration mode. You can configure the **source template** command only once to an interface.

Beginning in privileged EXEC mode, follow these steps to configure ACL in a template:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **ip access-list extended** *name*<br>**Example:**<br>`Device(config)# ip access-list extended 150` | Defines an extended IPv4 access list using a name, and enter access-list configuration mode.<br>Enter *name* to define access-list name.<br>Enter *number* to define extended IP access-list number. The range is from 100 to 199.<br>Enter *ext_number* to define extended IP access-list number. The expanded range is from 2000 to 2699. |
| **Step 4** | **ip access-list extended** {*name*\|*number*\|*ext_number*}<br>**Example:**<br>`Device(config)# ip access-list extended 151` | Defines an extended IPv4 access list using a name, and enter access-list configuration mode.<br>Enter *name* to define access-list name.<br>Enter *number* to define extended IP access-list number. The range is from 100 to 199.<br>Enter *ext_number* to define extended IP access-list number. The expanded range is from 2000 to 2699. |
| **Step 5** | **exit**<br>**Example:**<br>`Device(config-ext-nacl)# exit` | Exits access-list configuration mode. |
| **Step 6** | **template**<br>**Example:**<br>`Device# template test` | Creates a user template and enters template configuration mode. |
| **Step 7** | **ip access-group** {*access-list-number* \| *name*} {**in** \| **out**}<br>**Example:**<br>`Device(config-template)# ip access-group 150 in` | Controls access to the specified interface.<br>Enter *access-list-number* to define the access list. The access list can be a number.<br>Enter *name* to define the access list. The access list can be a name.<br>Enter **in** to direct the access list in the incoming direction of the interface.<br>Enter **out** to direct the access list in the outgoing direction of the interface. |
| **Step 8** | **exit**<br>**Example:**<br>`Device(config-template)# exit` | Exits template configuration mode and returns to privileged EXEC mode. |
| **Step 9** | **interface** *interface-id*<br>**Example:** | Identifies a specific interface for configuration, and enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# interface gigabitethernet1/0/1` | The interface can be a Layer 2 interface (port ACL), or a Layer 3 interface (router ACL). |
| **Step 10** | **ip access-group** {*access-list-number* \| *name*} {**in** \| **out**}<br>**Example:**<br>`Device(config-if)# ip access-group 151 out` | Controls access to the specified interface.<br>Enter *access-list-number* to define the access list. The access list can be a number.<br>Enter *name* to define the access list. The access list can be a name.<br>Enter **in** to direct the access list in the incoming direction of the interface.<br>Enter **out** to direct the access list in the outgoing direction of the interface. |
| **Step 11** | **source template** *name*<br>**Example:**<br>`Device(config)# source template test` | Applies an interface template to a target. The accesss list 150 is the incoming access list that is configured. |
| **Step 12** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring VLAN Maps

To create a VLAN map and apply it to one or more VLANs, perform these steps:

### Before you begin

Create the standard or extended IPv4 ACLs or named MAC extended ACLs that you want to apply to the VLAN.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **vlan access-map** *name* [**number**]<br>**Example:**<br>`Device(config)# vlan access-map map1 20` | Creates a VLAN map, and give it a name and (optionally) a number. The number is the sequence number of the entry within the map.<br>When you create VLAN maps with the same name, numbers are assigned sequentially in increments of 10. When modifying or deleting maps, you can enter the number of the map entry that you want to modify or delete.<br>VLAN maps do not use the specific permit or deny keywords. To deny a packet by using VLAN maps, create an ACL that would match the packet, and set the action to drop. A permit in the ACL counts as a match. A deny in the ACL means no match.<br>Entering this command changes to access-map configuration mode. |
| **Step 4** | **match** {**ip** \| **mac**} **address** {*name* \| *number*} [*name* \| *number*]<br>**Example:**<br>`Device(config-access-map)# match ip address ip2` | Match the packet (using either the IP or MAC address) against one or more standard or extended access lists. Note that packets are only matched against access lists of the correct protocol type. IP packets are matched against standard or extended IP access lists. Non-IP packets are only matched against named MAC extended access lists.<br>**Note**<br>If the VLAN map is configured with a match clause for a type of packet (IP or MAC) and the map action is drop, all packets that match the type are dropped. If the VLAN map has no match clause, and the configured action is drop, all IP and Layer 2 packets are dropped. |
| **Step 5** | Enter one of the following commands to specify an IP packet or a non-IP packet (with only a known MAC address) and to match the packet against one or more ACLs (standard or extended):<br>• **action** { **forward**}<br>`Device(config-access-map)# action forward`<br>• **action** { **drop**}<br>`Device(config-access-map)# action drop` | Sets the action for the map entry. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **exit**<br>**Example:**<br>`Device(config-access-map)# exit` | Exits access-map configuration mode. and returns to global configuration mode. |
| **Step 7** | **vlan filter** *mapname* **vlan-list** *list*<br>**Example:**<br>`Device(config)# vlan filter map1 vlan-list 20-22` | Applies the VLAN map to one or more VLAN IDs.<br>The list can be a single VLAN ID (22), a consecutive list (10-22), or a string of VLAN IDs (12, 22, 30). Spaces around the comma and hyphen are optional. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Applying a VLAN Map to a VLAN

To apply a VLAN map to one or more VLANs, perform these steps.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **vlan filter** *mapname* **vlan-list** *list*<br>**Example:**<br>`Device(config)# vlan filter map 1 vlan-list 20-22` | Applies the VLAN map to one or more VLAN IDs.<br>The list can be a single VLAN ID (22), a consecutive list (10-22), or a string of VLAN IDs (12, 22, 30). Spaces around the comma and hyphen are optional. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Monitoring IPv4 ACLs

You can monitor IPv4 ACLs by displaying the ACLs that are configured on the device, and displaying the ACLs that have been applied to interfaces and VLANs.

When you use the **ip access-group** interface configuration command to apply ACLs to a Layer 2 or 3 interface, you can display the access groups on the interface. You can also display the MAC ACLs applied to a Layer 2 interface. You can use the privileged EXEC commands as described in this table to display this information.

***Table 22: Commands for Displaying Access Lists and Access Groups***

| Command | Purpose |
|---|---|
| **show access-lists** [*number* \| *name*] | Displays the contents of one or all current IP and MAC address a specific access list (numbered or named). |
| **show ip access-lists** [*number* \| *name*] | Displays the contents of all current IP access lists or a specific I (numbered or named). |
| **show ip interface** *interface-id* | Displays detailed configuration and status of an interface. If IP the interface and ACLs have been applied by using the **ip access-g** configuration command, the access groups are included in the d |
| **show running-config** [**interface** *interface-id*] | Displays the contents of the configuration file for the device or interface, including all configured MAC and IP access lists and groups are applied to an interface. |
| **show mac access-group** [**interface** *interface-id*] | Displays MAC access lists applied to all Layer 2 interfaces or th Layer 2 interface. |

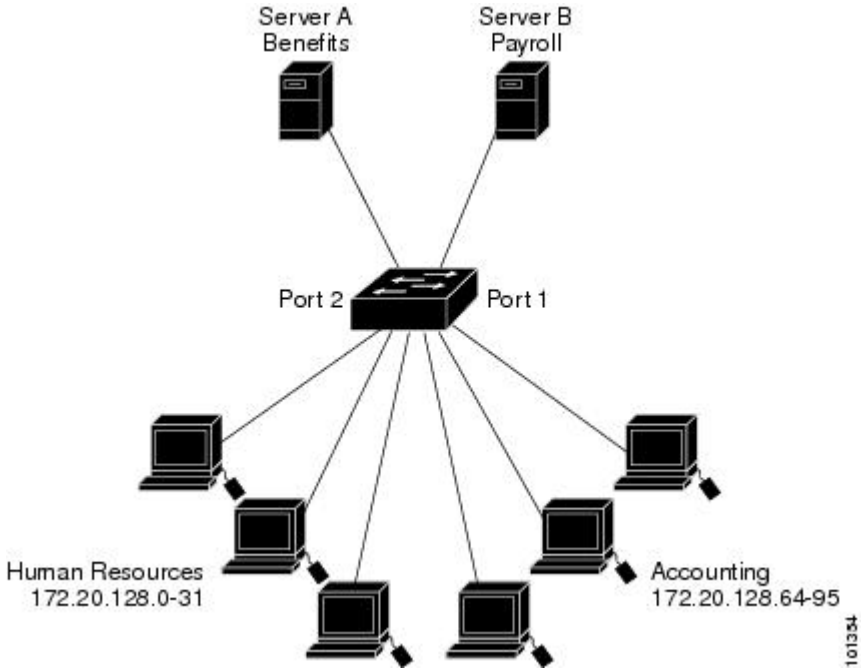
# Configuration Examples for IPv4 Access Control Lists

## ACLs in a Small Networked Office

***Figure 17: Using Router ACLs to Control Traffic***

This shows a small networked office environment with routed Port 2 connected to Server A, containing benefits and other information that all employees can access, and routed Port 1 connected to Server B, containing

confidential payroll data. All users can access Server A, but Server B has restricted access.




Use router ACLs to do this in one of two ways:

- Create a standard ACL, and filter traffic coming to the server from Port 1.
- Create an extended ACL, and filter traffic coming from the server into Port 1.

# Examples: ACLs in a Small Networked Office

This example uses a standard ACL to filter traffic coming into Server B from a port, permitting traffic only from Accounting's source addresses 172.20.128.64 to 172.20.128.95. The ACL is applied to traffic coming out of routed Port 1 from the specified source address.

```
Device> enable
Device# configure terminal
Device(config)# access-list 6 permit 172.20.128.64 0.0.0.31
Device(config)# exit
Device# show access-lists

Standard IP access list 6
    10 permit 172.20.128.64, wildcard bits 0.0.0.31

Device# configure terminal
Device(config)# interface gigabitethernet1/0/1
Device(config-if)# ip access-group 6 out
Device(config-if)# end
```

This example uses an extended ACL to filter traffic coming from Server B into a port, permitting traffic from any source address (in this case Server B) to only the Accounting destination addresses 172.20.128.64 to 172.20.128.95. The ACL is appliDeviceed to traffic going into routed Port 1, permitting it to go only to the specified destination addresses. Note that with extended ACLs, you must enter the protocol (IP) before the source and destination information.

```
Device(config)# access-list 106 permit ip any 172.20.128.64 0.0.0.31
Device(config)# exit
Device# show access-lists

Extended IP access list 106
    10 permit ip any 172.20.128.64 0.0.0.31


Device# configure terminal
Device(config)# interface gigabitethernet1/0/1
Device(config-if)# ip access-group 106 in
Device(config-if)# end
```

# Example: Numbered ACLs

In this example, network 10.0.0.0 is a Class A network whose second octet specifies a subnet; that is, its subnet mask is 255.255.0.0. The third and fourth octets of a network 10.0.0.0 address specify a particular host. Using access list 2, the switch accepts one address on subnet 48 and reject all others on that subnet. The last line of the list shows that the switch accepts addresses on all other network 10.0.0.0 subnets. The ACL is applied to packets entering a port.

```
Device> enable
Device# configure terminal
Device(config)# access-list 2 permit 10.48.0.3
Device(config)# access-list 2 deny 10.48.0.0 0.0.255.255
Device(config)# access-list 2 permit 10.0.0.0 0.255.255.255
Device(config)# interface gigabitethernet2/0/1
Device(config-if)# ip access-group 2 in
Device(config-if)# end
```

# Examples: Extended ACLs

In this example, the first line permits any incoming TCP connections with destination ports greater than 1023. The second line permits incoming TCP connections to the Simple Mail Transfer Protocol (SMTP) port of host 172.16.0.0. The third line permits incoming ICMP messages for error feedback.

```
Device> enable
Device# configure terminal
Device(config)# access-list 102 permit tcp any 172.16.0.0 0.0.255.255 gt 1023
Device(config)# access-list 102 permit tcp any host 172.16.1.2 eq 25
Device(config)# access-list 102 permit icmp any any
Device(config)# interface gigabitethernet 2/0/1
Device(config-if)# ip access-group 102 in
Device(config-if)# end
```

In this example, suppose that you have a network connected to the Internet, and you want any host on the network to be able to form TCP connections to any host on the Internet. However, you do not want IP hosts to be able to form TCP connections to hosts on your network, except to the mail (SMTP) port of a dedicated mail host.

SMTP uses TCP port 25 on one end of the connection and a random port number on the other end. The same port numbers are used throughout the life of the connection. Mail packets coming in from the Internet have a destination port of 25. Because the secure system of the network always accepts mail connections on port 25, the incoming sevices are separately controlled.

```
Device> enable
Device# configure terminal
Device(config)# access-list 102 permit tcp any 172.16.0.0 0.0.255.255 eq 23
Device(config)# access-list 102 permit tcp any 172.16.0.0 0.0.255.255 eq 25
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# ip access-group 102 in
Device(config-if)# end
```

In this example, the network is a Class B network with the address 172.16.0.0, and the mail host address is 172.16.1.2. The **established** keyword is used only for the TCP to show an established connection. A match occurs if the TCP datagram has the ACK or RST bits set, which show that the packet belongs to an existing connection. Gigabit Ethernet interface 1 is the interface that connects the device to the Internet.

```
Device> enable
Device# configure terminal
Device(config)# access-list 102 permit tcp any 172.16.0.0 0.0.255.255 established
Device(config)# access-list 102 permit tcp any host 172.16.1.2 eq 25
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# ip access-group 102 in
Device(config-if)# end
```

# Examples: Named ACLs

### Creating named standard and extended ACLs

This example creates a standard ACL named *internet_filter* and an extended ACL named *marketing_group*. The *internet_filter* ACL allows all traffic from the source address 10.2.3.4.

```
Device> enable
Device# configure terminal
Device(config)# ip access-list standard Internet_filter
Device(config-ext-nacl)# permit 10.2.3.4
Device(config-ext-nacl)# exit
Device(config-ext-nacl)# end
```

The *marketing_group* ACL allows any TCP Telnet traffic to the destination address and wildcard 172.16.0.0 0.0.255.255 and denies any other TCP traffic. It permits ICMP traffic, denies UDP traffic from any source to the destination address range 172.16.0.0 through 172.16.255.255 with a destination port less than 1024, denies any other IP traffic, and provides a log of the result.

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended marketing_group
Device(config-ext-nacl)# permit tcp any 172.16.0.0 0.0.255.255 eq telnet
Device(config-ext-nacl)# deny tcp any any
Device(config-ext-nacl)# permit icmp any any
Device(config-ext-nacl)# deny udp any 172.16.0.0 0.0.255.255 lt 1024
Device(config-ext-nacl)# deny ip any any log
Device(config-ext-nacl)# end
```

The *Internet_filter* ACL is applied to outgoing traffic and the *marketing_group* ACL is applied to incoming traffic on a Layer 3 port.

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet3/0/2
Device(config-if)# no switchport
Device(config-if)# ip address 10.0.5.1 255.255.255.0
```

```
Device(config-if)# ip access-group Internet_filter out
Device(config-if)# ip access-group marketing_group in
Device(config-if)# end
```

### Deleting Individual ACEs from Named ACLs

This example shows how to delete individual ACEs from the named access list *border-list*:

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended border-list
Device(config-ext-nacl)# no permit ip host 10.1.1.3 any
Device(config-ext-nacl)# end
```

## Examples: ACL Logging

Two variations of logging are supported on router ACLs. The **log** keyword sends an informational logging message to the console about the packet that matches the entry; the **log-input** keyword includes the input interface in the log entry.

In this example, standard named access list *stan1* denies traffic from 10.1.1.0 0.0.0.255, allows traffic from all other sources, and includes the **log** keyword.

```
Device> enable
Device# configure terminal
Device(config)# ip access-list standard stan1
Device(config-std-nacl)# deny 10.1.1.0 0.0.0.255 log
Device(config-std-nacl)# permit any log
Device(config-std-nacl)# exit
Device(config)# interface gigabitethernet1/0/1
Device(config-if)# ip access-group stan1 in
Device(config-if)# end
Device# show logging

Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
    Console logging: level debugging, 37 messages logged
    Monitor logging: level debugging, 0 messages logged
    Buffer logging: level debugging, 37 messages logged
    File logging: disabled
    Trap logging: level debugging, 39 message lines logged

Log Buffer (4096 bytes):

00:00:48: NTP: authentication delay calculation problems

<output truncated>

00:09:34:%SEC-6-IPACCESSLOGS:list stan1 permitted 0.0.0.0 1 packet
00:09:59:%SEC-6-IPACCESSLOGS:list stan1 denied 10.1.1.15 1 packet
00:10:11:%SEC-6-IPACCESSLOGS:list stan1 permitted 0.0.0.0 1 packet
```

This example is a named extended access list *ext1* that permits ICMP packets from any source to 10.1.1.0 0.0.0.255 and denies all UDP packets.

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended ext1
Device(config-ext-nacl)# permit icmp any 10.1.1.0 0.0.0.255 log
Device(config-ext-nacl)# deny udp any any log
Device(config-std-nacl)# exit
Device(config)# interface gigabitethernet1/0/2
```

```
Device(config-if)# ip access-group ext1 in
Device(config)# end
```

This is a an example of a log for an extended ACL:

```
01:24:23:%SEC-6-IPACCESSLOGDP:list ext1 permitted icmp 10.1.1.15 -> 10.1.1.61 (0/0), 1
packet
01:25:14:%SEC-6-IPACCESSLOGDP:list ext1 permitted icmp 10.1.1.15 -> 10.1.1.61 (0/0), 7
packets
01:26:12:%SEC-6-IPACCESSLOGP:list ext1 denied udp 0.0.0.0(0) -> 255.255.255.255(0), 1 packet
01:31:33:%SEC-6-IPACCESSLOGP:list ext1 denied udp 0.0.0.0(0) -> 255.255.255.255(0), 8 packets
```

Note that all logging entries for IP ACLs start with %SEC-6-IPACCESSLOG with minor variations in format depending on the kind of ACL and the access entry that has been matched.

This is an example of an output message when the **log-input** keyword is entered:

```
00:04:21:%SEC-6-IPACCESSLOGDP:list inputlog permitted icmp 10.1.1.10 (Vlan1 0001.42ef.a400)
 ->
10.1.1.61 (0/0), 1 packet
```

A log message for the same sort of packet using the **log** keyword does not include the input interface information:

```
00:05:47:%SEC-6-IPACCESSLOGDP:list inputlog permitted icmp 10.1.1.10 -> 10.1.1.61 (0/0), 1
 packet
```

# Example: ACEs and Fragmented and Unfragmented Traffic

Consider access list 102, configured with these commands, applied to three fragmented packets:

```
Device> enable
Device# configure terminal
Device(config)# access-list 102 permit tcp any host 10.1.1.1 eq smtp
Device(config)# access-list 102 deny tcp any host 10.1.1.2 eq telnet
Device(config)# access-list 102 permit tcp any host 10.1.1.2
Device(config)# access-list 102 deny tcp any any
Device(config)# end
```

**Note** In the first and second ACEs in the examples, the *eq* keyword after the destination address means to test for the TCP-destination-port well-known numbers equaling Simple Mail Transfer Protocol (SMTP) and Telnet, respectively.

- Packet A is a TCP packet from host 10.2.2.2., port 65000, going to host 10.1.1.1 on the SMTP port. If this packet is fragmented, the first fragment matches the first ACE (a permit) as if it were a complete packet because all Layer 4 information is present. The remaining fragments also match the first ACE, even though they do not contain the SMTP port information, because the first ACE only checks Layer 3 information when applied to fragments. The information in this example is that the packet is TCP and that the destination is 10.1.1.1.
- Packet B is from host 10.2.2.2, port 65001, going to host 10.1.1.2 on the Telnet port. If this packet is fragmented, the first fragment matches the second ACE (a deny) because all Layer 3 and Layer 4 information is present. The remaining fragments in the packet do not match the second ACE because they are missing Layer 4 information. Instead, they match the third ACE (a permit).

Because the first fragment was denied, host 10.1.1.2 cannot reassemble a complete packet, so packet B is effectively denied. However, the later fragments that are permitted will consume bandwidth on the network and resources of host 10.1.1.2 as it tries to reassemble the packet.

- Fragmented packet C is from host 10.2.2.2, port 65001, going to host 10.1.1.3, port ftp. If this packet is fragmented, the first fragment matches the fourth ACE (a deny). All other fragments also match the fourth ACE because that ACE does not check any Layer 4 information and because Layer 3 information in all fragments shows that they are being sent to host 10.1.1.3, and the earlier permit ACEs were checking different hosts.

# Examples: Using Time Ranges with ACLs

This example shows how to verify after you configure time ranges for *workhours* and to configure January 1, 2006, as a company holiday.

```
Device# show time-range

time-range entry: new_year_day_2003 (inactive)
   absolute start 00:00 01 January 2006 end 23:59 01 January 2006
time-range entry: workhours (inactive)
   periodic weekdays 8:00 to 12:00
   periodic weekdays 13:00 to 17:00
```

To apply a time range, enter the time-range name in an extended ACL that can implement time ranges. This example shows how to create and verify extended access list 188 that denies TCP traffic from any source to any destination during the defined holiday times and permits all TCP traffic during work hours.

```
Device> enable
Device# configure terminal
Device(config)# access-list 188 deny tcp any any time-range new_year_day_2006
Device(config)# access-list 188 permit tcp any any time-range workhours
Device(config)# exit
Device# show access-lists

Extended IP access list 188
   10 deny tcp any any time-range new_year_day_2006 (inactive)
   20 permit tcp any any time-range workhours (inactive)
```

This example uses named ACLs to permit and deny the same traffic.

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended deny_access
Device(config-ext-nacl)# deny tcp any any time-range new_year_day_2006
Device(config-ext-nacl)# exit
Device(config)# ip access-list extended may_access
Device(config-ext-nacl)# permit tcp any any time-range workhours
Device(config-ext-nacl)# end
Device# show ip access-lists

Extended IP access list lpip_default
    10 permit ip any any
Extended IP access list deny_access
    10 deny tcp any any time-range new_year_day_2006 (inactive)
Extended IP access list may_access
    10 permit tcp any any time-range workhours (inactive)
```

# Examples: Time Range Applied to an IP ACL

This example denies HTTP traffic on IP on Monday through Friday between the hours of 8:00 a.m. and 6:00 p.m (18:00). The example allows UDP traffic only on Saturday and Sunday from noon to 8:00 p.m. (20:00).

```
Device> enable
Device# configure terminal
Device(config)# time-range no-http
Device(config)# periodic weekdays 8:00 to 18:00
Device(config)# time-range udp-yes
Device(config)# periodic weekend 12:00 to 20:00
Device(config)# ip access-list extended strict
Device(config-ext-nacl)# deny tcp any any eq www time-range no-http
Device(config-ext-nacl)# permit udp any any time-range udp-yes
Device(config-ext-nacl)# exit
Device(config)# interface gigabitethernet2/0/1
Device(config-if)# ip access-group strict in
Device(config-if)# end
```

# Examples: Including Comments in ACLs

You can use the **remark** keyword to include comments (remarks) about entries in any IP standard or extended ACL. The remarks make the ACL easier for you to understand and scan. Each remark line is limited to 100 characters.

The remark can go before or after a permit or deny statement. You should be consistent about where you put the remark so that it is clear which remark describes which permit or deny statement. For example, it would be confusing to have some remarks before the associated permit or deny statements and some remarks after the associated statements.

To include a comment for IP numbered standard or extended ACLs, use the **access-list** *access-list number* **remark** *remark* global configuration command. To remove the remark, use the **no** form of this command.

In this example, the workstation that belongs to user1 is allowed access, and the workstation that belongs to user2 is not allowed access:

```
Device> enable
Device# configure terminal
Device(config)# access-list 1 remark Permit only user1 workstation through
Device(config)# access-list 1 permit 171.69.2.88
Device(config)# access-list 1 remark Do not allow user2 through
Device(config)# access-list 1 deny 171.69.3.13
Device(config)# end
```

For an entry in a named IP ACL, use the **remark** access-list configuration command. To remove the remark, use the **no** form of this command.

In this example, the subnet1 subnet is not allowed to use outbound Telnet:

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended telnetting
Device(config-ext-nacl)# remark Do not allow subnet1 subnet to telnet out
Device(config-ext-nacl)# deny tcp host 171.69.2.88 any eq telnet
Device(config-ext-nacl)# end
```

# Example: Creating an ACL and a VLAN Map to Deny a Packet

This example shows how to create an ACL and a VLAN map to deny a packet. In the first map, any packets that match the *ip1* ACL (TCP packets) would be dropped. You first create the *ip1* ACL to permit any TCP packet and no other packets. Because there is a match clause for IP packets in the VLAN map, the default action is to drop any IP packet that does not match any of the match clauses.

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended ip1
Device(config-ext-nacl)# permit tcp any any
Device(config-ext-nacl)# exit
Device(config)# vlan access-map map_1 10
Device(config-access-map)# match ip address ip1
Device(config-access-map)# action drop
Device(config-access-map)# end
```

# Example: Creating an ACL and a VLAN Map to Permit a Packet

This example shows how to create a VLAN map to permit a packet. ACL *ip2* permits UDP packets and any packets that match the *ip2* ACL are forwarded. In this map, any IP packets that did not match any of the previous ACLs (that is, packets that are not TCP packets or UDP packets) would get dropped.

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended ip2
Device(config-ext-nacl)# permit udp any any
Device(config-ext-nacl)# exit
Device(config)# vlan access-map map_1 20
Device(config-access-map)# match ip address ip2
Device(config-access-map)# action forward
Device(config-access-map)# exit
```

# Example: Default Action of Dropping IP Packets and Forwarding MAC Packets

In this example, the VLAN map has a default action of drop for IP packets and a default action of forward for MAC packets. Used with standard ACL 101 and extended named access lists **igmp-match** and **tcp-match**, the map will have the following results:

- Forward all UDP packets
- Drop all IGMP packets
- Forward all TCP packets
- Drop all other IP packets
- Forward all non-IP packets

```
Device> enable
Device# configure terminal
Device(config)# access-list 101 permit udp any any
Device(config)# ip access-list extended igmp-match
Device(config-ext-nacl)# permit igmp any any
Device(config)# action forward
Device(config-ext-nacl)# permit tcp any any
Device(config-ext-nacl)# exit
Device(config)# vlan access-map drop-ip-default 10
```

```
Device(config-access-map)# match ip address 101
Device(config-access-map)# action forward
Device(config-access-map)# exit
Device(config)# vlan access-map drop-ip-default 20
Device(config-access-map)# match ip address igmp-match
Device(config-access-map)# action drop
Device(config-access-map)# exit
Device(config)# vlan access-map drop-ip-default 30
Device(config-access-map)# match ip address tcp-match
Device(config-access-map)# action forward
Device(config-access-map)# end
```

# Example: Default Action of Dropping MAC Packets and Forwarding IP Packets

In this example, the VLAN map has a default action of drop for MAC packets and a default action of forward for IP packets. Used with MAC extended access lists **good-hosts** and **good-protocols**, the map will have the following results:

- Forward MAC packets from hosts 0000.0c00.0111 and 0000.0c00.0211
- Forward MAC packets with decnet-iv or vines-ip protocols
- Drop all other non-IP packets
- Forward all IP packets

```
Device> enable
Device# configure terminal
Device(config)# mac access-list extended good-hosts
Device(config-ext-macl)# permit host 000.0c00.0111 any
Device(config-ext-macl)# permit host 000.0c00.0211 any
Device(config-ext-nacl)# exit
Device(config)# action forward
Device(config-ext-macl)# mac access-list extended good-protocols
Device(config-ext-macl)# permit any any vines-ip
Device(config-ext-nacl)# exit
Device(config)# vlan access-map drop-mac-default 10
Device(config-access-map)# match mac address good-hosts
Device(config-access-map)# action forward
Device(config-access-map)# exit
Device(config)# vlan access-map drop-mac-default 20
Device(config-access-map)# match mac address good-protocols
Device(config-access-map)# action forward
Device(config-access-map)# end
```

# Example: Default Action of Dropping All Packets

In this example, the VLAN map has a default action of drop for all packets (IP and non-IP). Used with access lists **tcp-match** and **good-hosts** from Examples 2 and 3, the map will have the following results:

- Forward all TCP packets
- Forward MAC packets from hosts 0000.0c00.0111 and 0000.0c00.0211
- Drop all other IP packets
- Drop all other MAC packets

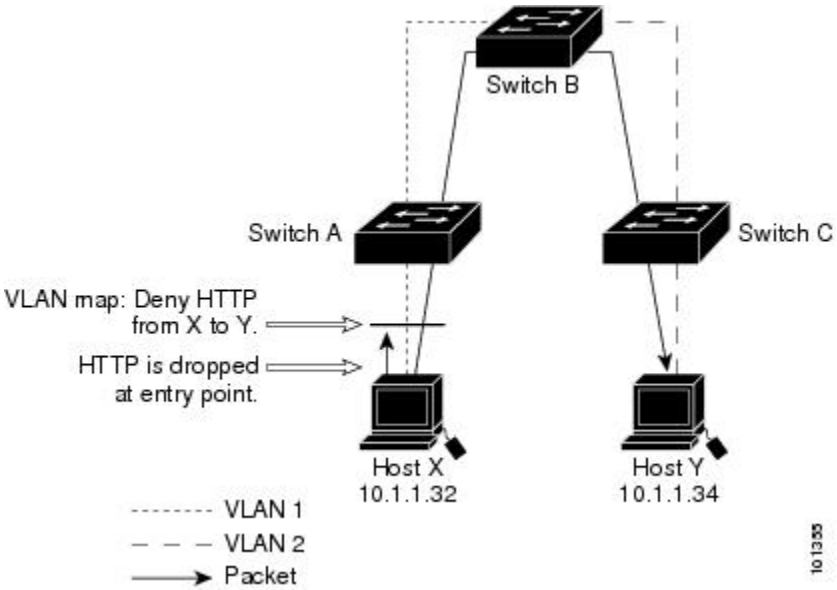
```
Device> enable
Device# configure terminal
Device(config)# vlan access-map drop-all-default 10
Device(config-access-map)# match ip address tcp-match
Device(config-access-map)# action forward
Device(config-access-map)# exit
Device(config)# vlan access-map drop-all-default 20
Device(config-access-map)# match mac address good-hosts
Device(config-access-map)# action forward
Device(config-access-map)# end
```

# Example: Using VLAN Maps in a Network

## Example: Wiring Closet Configuration

***Figure 18: Wiring Closet Configuration***

In a wiring closet configuration, routing might not be enabled on the switch. In this configuration, the switch can still support a VLAN map and a QoS classification ACL. Assume that Host X and Host Y are in different VLANs and are connected to wiring closet switches A and C. Traffic from Host X to Host Y is eventually being routed by Switch B, a Layer 3 switch with routing enabled. Traffic from Host X to Host Y can be access-controlled at the traffic entry point, Switch A.




If you do not want HTTP traffic switched from Host X to Host Y, you can configure a VLAN map on Switch A to drop all HTTP traffic from Host X (IP address 10.1.1.32) to Host Y (IP address 10.1.1.34) at Switch A and not bridge it to Switch B.

First, define the IP access list *http* that permits (matches) any TCP traffic on the HTTP port.

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended http
Device(config-ext-nacl)# permit tcp host 10.1.1.32 host 10.1.1.34 eq www
Device(config-ext-nacl)# end
```

Next, create VLAN access map *map2* so that traffic that matches the *http* access list is dropped and all other IP traffic is forwarded.

```
Device> enable
Device# configure terminal
Device(config)# vlan access-map map2 10
Device(config-access-map)# match ip address http
Device(config-access-map)# action drop
Device(config-access-map)# exit
Device(config)# ip access-list extended match_all
Device(config-ext-nacl)# permit ip any any
Device(config-ext-nacl)# exit
Device(config)# vlan access-map map2 20
Device(config-access-map)# match ip address match_all
Device(config-access-map)# action forward
Device(config-access-map)# end
```

Then, apply VLAN access map *map2* to VLAN 1.

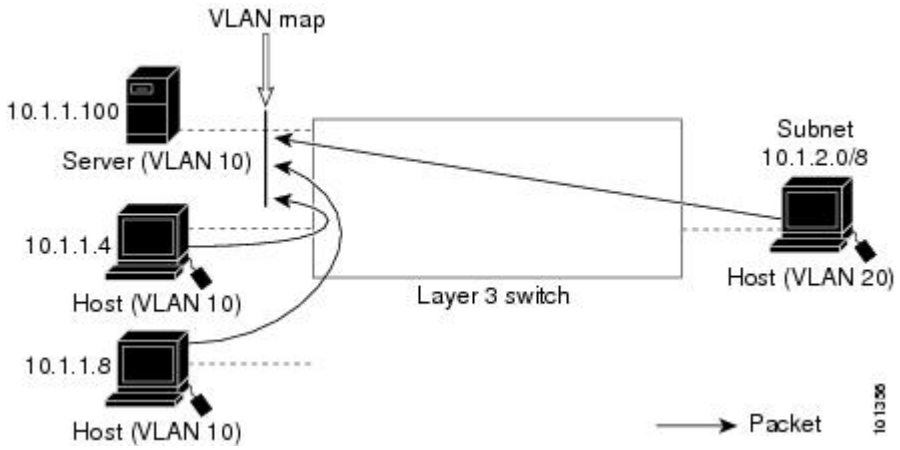
```
Device> enable
Device# configure terminal
Device(config)# vlan filter map2 vlan 1
Device(config)# end
```

## Example: Restricting Access to a Server on Another VLAN

***Figure 19: Restricting Access to a Server on Another VLAN***

You can restrict access to a server on another VLAN. For example, server 10.1.1.100 in VLAN 10 needs to have access denied to these hosts:

- Hosts in subnet 10.1.2.0/8 in VLAN 20 should not have access.
- Hosts 10.1.1.4 and 10.1.1.8 in VLAN 10 should not have access.




## Example: Denying Access to a Server on Another VLAN

This example shows how to deny access to a server on another VLAN by creating the VLAN map SERVER 1 that denies access to hosts in subnet 10.1.2.0.8, host 10.1.1.4, and host 10.1.1.8 and permits other IP traffic. The final step is to apply the map SERVER1 to VLAN 10.

Define the IP ACL that will match the correct packets.

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended SERVER1_ACL
```

```
Device(config-ext-nacl)# permit ip 10.1.2.0 0.0.0.255 host 10.1.1.100
Device(config-ext-nacl)# permit ip host 10.1.1.4 host 10.1.1.100
Device(config-ext-nacl)# permit ip host 10.1.1.8 host 10.1.1.100
Device(config-ext-nacl)# end
```

Define a VLAN map using this ACL that will drop IP packets that match SERVER1_ACL and forward IP packets that do not match the ACL.

```
Device> enable
Device# configure terminal
Device(config)# vlan access-map SERVER1_MAP
Device(config-access-map)# match ip address SERVER1_ACL
Device(config-access-map)# action drop
Device(config)# vlan access-map SERVER1_MAP 20
Device(config-access-map)# action forward
Device(config-access-map)# end
```

Apply the VLAN map to VLAN 10.

```
Device> enable
Device# configure terminal
Device(config)# vlan filter SERVER1_MAP vlan-list 10
Device(config)# end
```

# Additional References for IPv4 Access Control Lists

### Related Documents

| Related Topic | Document Title |
|---|---|
| IPv6 ACLs | IPv6 ACLs chapter of the *Security Configuration Guide* |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/support |

# Feature History for IPv4 Access Control Lists

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | IPv4 Access Control Lists | This chapter describes how to configure network security on the switch by using ACLs. Packet filtering can help limit network traffic and restrict network use by certain users or devices. ACLs filter traffic as it passes through device and permit or deny packets crossing specified interfaces.<br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | IPv4 Access Control Lists | Support for this feature was introduced on the C9500-24Y4C, C9500-32C, C9500-32QC, and C9500-48Y4C models of the Cisco Catalyst 9500 Series Switches. |

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Bengaluru 17.5.1 | ACL template support for IPv4 | Interface template allows you to configure multiple commands and associate it with an interface. The **ip access-group** command is used to apply an IPv4 access list under template mode of configuration. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 22

# IPv6 ACLs


- Restrictions for IPv6 ACLs, on page 385
- Information About IPv6 ACLs, on page 386
- How to Configure an IPv6 ACL, on page 388
- Monitoring IPv6 ACLs, on page 397
- Configuration Examples for IPv6 ACL, on page 397
- Feature History for IPv6 ACLs, on page 398


## Restrictions for IPv6 ACLs

IPv6 supports only named ACLs. With IPv4 ACLs, you can configure standard and extended numbered IP ACLs, named IP ACLs, and MAC ACLs.

The switch supports most Cisco IOS-supported IPv6 ACLs with some exceptions:

- The switch does not support matching on these keywords: **flowlabel**, **routing header**, and **undetermined-transport**.
- The switch does not support reflexive ACLs (the **reflect** keyword).
- The switch does not apply MAC-based ACLs on IPv6 frames.
- When configuring an ACL, there is no restriction on keywords that are entered in the ACL, regardless of whether they are supported or not on the platform. When you apply the ACL to an interface that requires hardware forwarding (physical ports or SVIs), the switch checks to determine whether ACL can be supported on the interface or not. If the ACL is not supported on the interface, the ACL is rejected.
- If an ACL is applied to an interface and you attempt to add an access control entry (ACE) with an unsupported keyword, the switch does not allow the ACE to be added to the ACL that is currently attached to the interface.
- When you apply a scale ACL to an interface that does not program TCAM for a protocol and the ACLs that have been unloaded, it can impact the existing normal movement of traffic for other protocols. The restriction is applicable to IPv6 and MAC address traffic for C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series SwitchesIPv4, IPv6, and MAC address traffic for C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches.
- Time-to-live (TTL) classification is not supported on ACLs.

- If a downloadable ACL contains any type of duplicate entries, the entries are not auto merged. As a result, the 802.1X session authorization fails. Ensure that the downloadable ACL is optimized without any duplicate entries, for example port-based and name-based entries for the same port.

# Information About IPv6 ACLs

The following sections provide information about IPv6 ACLs.

## IPv6 ACL Overview

This topic provides an overview of IPv6 ACL.

An access control list (ACL) is a set of rules that are used to limit access to a particular interface. ACLs are configured on the device and applied to the management interface and to any of the dynamic interfaces.

You can also create a preauthentication ACL for web authentication. Such an ACL is used to allow certain types of traffic before authentication is complete.

IPv6 ACLs support the same options as IPv4 ACLs including source, destination, source, and destination ports.

## Supported ACLs

The switch supports three types of ACLs to filter the traffic:

- Port ACLs access-control traffic entering a Layer 2 interface. You can apply port ACLs to a Layer 2 interface in each direction to each access list type—IPv4 and MAC.
- Router ACLs access-control traffic routed between VLANs and are applied to Layer 3 interfaces in a specific direction (inbound or outbound).
- VLAN ACLs or VLAN maps are applied only to Layer 2 VLANs and impact bridged traffic only. You can use VLAN maps to filter traffic between devices in the same VLAN. VLAN maps are configured to provide access control based on Layer 3 addresses for IPv4. Unsupported protocols are access-controlled through MAC addresses using Ethernet ACEs. After a VLAN map is applied to a VLAN, all packets (routed or bridged) entering the VLAN are checked against the VLAN map. Packets can either enter the VLAN through a switch port or through a routed port after being routed.

## Types of ACL

The following sections provide information on the types of ACL:

### Per-User IPv6 ACL

For the per-user ACL, the full access control entries (ACE) as the text strings are configured on the Cisco Secure Access Control Server (Cisco Secure ACS).

## Filter ID IPv6 ACL

For the filter-Id ACL, the full ACEs and the `acl name(filter-id)` is configured on the device and only the `filter-id` is configured on the Cisco Secure ACS.

## Downloadable IPv6 ACL

For the downloadable ACL (dACL), all the full ACEs and the dacl name are configured only on the Cisco Secure ACS.

The Cisco Secure ACS sends the dacl name to the device in its `ACCESS-Accept` attribute, which takes the `dacl` name and sends the `dACL` name back to the Cisco Secure ACS for the ACEs, using the `ACCESS-request` attribute.

# ACL Precedence

When VLAN maps, Port ACLs, and router ACLs are configured on the same switch, the filtering precedence, from greatest to least for ingress traffic is port ACL, VLAN map, and then router ACL. For egress traffic, the filtering precedence is router ACL, VLAN map, and then port ACL.

The following examples describe simple use cases:

- When both an input port ACL and a VLAN map are applied, incoming packets that are received on ports with a port ACL applied are filtered by the port ACL. Other packets are filtered by the VLAN map
- When an input router ACL and input port ACL exist in a switch virtual interface (SVI), incoming packets that are received on ports to which a port ACL is applied are filtered by the port ACL. Incoming routed IP packets received on other ports are filtered by the router ACL. Other packets are not filtered.
- When an output router ACL and input port ACL exist in an SVI, incoming packets that are received on the ports to which a port ACL is applied are filtered by the port ACL. Outgoing routed IP packets are filtered by the router ACL. Other packets are not filtered.
- When a VLAN map, input router ACL, and input port ACL exist in an SVI, incoming packets that are received on the ports to which a port ACL is applied are only filtered by the port ACL. Incoming routed IP packets received on other ports are filtered by both the VLAN map and the router ACL. Other packets are filtered only by the VLAN map.
- When a VLAN map, output router ACL, and input port ACL exist in an SVI, incoming packets that are received on the ports to which a port ACL is applied are only filtered by the port ACL. Outgoing routed IP packets are filtered by both the VLAN map and the router ACL. Other packets are filtered only by the VLAN map.

# VLAN Maps

VLAN ACLs or VLAN maps are used to control the network traffic within a VLAN. You can apply VLAN maps to all packets that are bridged within a VLAN in the switch or switch stack. VACLs are strictly for the security packet filtering and for redirecting traffic to specific physical interfaces. VACLs are not defined by direction (ingress or egress).

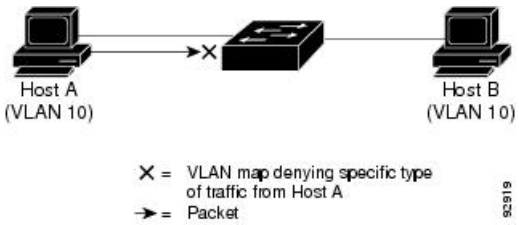
All non-IP protocols are access-controlled through MAC addresses and Ethertype using MAC VLAN maps. (IP traffic is not access-controlled by MAC VLAN maps.) You can enforce VLAN maps only on packets

going through the switch; you cannot enforce VLAN maps on traffic between hosts on a hub or on another switch that is connected to this switch.

With VLAN maps, forwarding of packets is permitted or denied, based on the action specified in the map.

***Figure 20: Using VLAN Maps to Control Traffic***

This figure shows how a VLAN map is applied to prevent a specific type of traffic from Host A in VLAN 10 from being forwarded. You can apply only one VLAN map to a VLAN.




## Interactions with Other Features and Switches

- If an IPv6 router ACL is configured to deny a packet, the packet is not routed. A copy of the packet is sent to the Internet Control Message Protocol (ICMP) queue to generate an ICMP unreachable message for the frame.
- If a bridged frame is to be dropped due to a port ACL, the frame is not bridged.
- You can create both IPv4 and IPv6 ACLs on a switch or switch stack, and you can apply both IPv4 and IPv6 ACLs to the same interface. Each ACL must have a unique name; an error message appears if you try to use a name that is already configured.

  You use different commands to create IPv4 and IPv6 ACLs and to attach IPv4 or IPv6 ACLs to the same Layer 2 or Layer 3 interface. If you use the wrong command to attach an ACL (for example, an IPv4 command to attach an IPv6 ACL), you receive an error message.
- You cannot use MAC ACLs to filter IPv6 frames. MAC ACLs can only filter non-IP frames.
- If the hardware memory is full, packets are dropped on the interface and an unload error message is logged.

  If the hardware memory is full, for any additional configured ACLs, packets are dropped to the CPU, and the ACLs are applied in software. When the hardware is full a message is printed to the console indicating the ACL has been unloaded and the packets will be dropped on the interface.

# How to Configure an IPv6 ACL

The following sections display information on how to configure an IPv6 ACL.

## Default Configuration for IPv6 ACLs

The default IPv6 ACL configuration is as follows:

```
Device# show access-lists preauth_ipv6_acl
```

```
IPv6 access list preauth_ipv6_acl (per-user)
permit udp any any eq domain sequence 10
permit tcp any any eq domain sequence 20
permit icmp any any nd-ns sequence 30
permit icmp any any nd-na sequence 40
permit icmp any any router-solicitation sequence 50
permit icmp any any router-advertisement sequence 60
permit icmp any any redirect sequence 70
permit udp any eq 547 any eq 546 sequence 80
permit udp any eq 546 any eq 547 sequence 90
deny ipv6 any any sequence 100
```

# Configuring IPv6 ACLs

To filter IPv6 traffic, perform this procedure:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ipv6 access-list** {*list-name* \| **log-update threshold** \| **role-based** *list-name*}<br>**Example:**<br>`Device(config)# ipv6 access-list example_acl_list` | Defines an IPv6 ACL name, and enters IPv6 access list configuration mode. |
| **Step 4** | {**deny** \| **permit**} protocol {*source-ipv6-prefix/ \|prefix-length* \|**any threshold**\| **host** *source-ipv6-address*} [ operator [ *port-number* ]] { *destination-ipv6-prefix/ prefix-length* \| **any** \| **host** *destination-ipv6-address*} [operator [*port-number*]][**dscp** *value*] [**fragments**] [**log**] [**log-input**][**sequence** *value*] [**time-range** *name*]<br>**Example:**<br>`Device(config-ipv6-acl)# permit tcp 2001:DB8:0300:0201::/32 eq telnet any` | Specifies permit or deny conditions for an IPv6 ACL.<br>• For protocol, enter the name or number of an IP: **ahp**, **esp**, **icmp**, **ipv6**, **pcp**, **stcp**, **tcp**, or **udp**, or an integer in the range 0 to 255 representing an IPv6 protocol number.<br>• The *source-ipv6-prefix/prefix-length* or *destination-ipv6-prefix/ prefix-length* is the source or destination IPv6 network or class of networks for which to set deny or permit conditions, specified in hexadecimal and using 16-bit values between colons (see RFC 2373).<br>• Enter **any** as an abbreviation for the IPv6 prefix ::/0. |

| | Command or Action | Purpose |
|---|---|---|
| | | • For **host** *source-ipv6-address* or *destination-ipv6-address*, enter the source or destination IPv6 host address for which to set deny or permit conditions, specified in hexadecimal using 16-bit values between colons.<br>• (Optional) For operator, specify an operand that compares the source or destination ports of the specified protocol. Operands are **lt** (less than), **gt** (greater than), **eq** (equal), **neq** (not equal), and **range**.<br>If the operator follows the *source-ipv6-prefix/prefix-length* argument, it must match the source port. If the operator follows the *destination-ipv6-prefix/prefix-length* argument, it must match the destination port.<br>• (Optional) The **port-number** is a decimal number from 0 to 65535 or the name of a TCP or UDP port. You can use TCP port names only when filtering TCP. You can use UDP port names only when filtering UDP.<br>• (Optional) Enter **dscp** value to match a differentiated services code point value against the traffic class value in the Traffic Class field of each IPv6 packet header. The acceptable range is from 0 to 63.<br>• (Optional) Enter **fragments** to check noninitial fragments. This keyword is visible only if the protocol is ipv6.<br>• (Optional) Enter **log** to cause an logging message to be sent to the console about the packet that matches the entry. Enter **log-input** to include the input interface in the log entry. Logging is supported only for router ACLs.<br>• (Optional) Enter **sequence** *value* to specify the sequence number for the access list statement. The acceptable range is from 1 to 4,294,967,295.<br>• (Optional) Enter **time-range** name to specify the time range that applies to the deny or permit statement. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | {**deny** \| **permit**} **tcp** {*source-ipv6-prefix/prefix-length* \| **any** \| **host** *source-ipv6-address*} [**operator** [**port-number**]] {*destination-ipv6-prefix/prefix-length* \| **any** \| **host** *destination-ipv6-address*} [operator [*port-number*]] [**ack**] [**dscp** *value*] [**established**] [**fin**] [**log**] [**log-input**] [**neq** {**port** \| protocol}] [**psh**] [**range** {**port** \| **protocol**}] [**rst**] [**sequence** *value*] [**syn**] [**time-range** *name*] [**urg**]<br>**Example:**<br>`Device(config-ipv6-acl)# deny tcp host 2001:DB8:1::1 any log-input` | Specifies permit or deny conditions for an IPv6 ACL.<br>Enter **tcp** for Transmission Control Protocol. The parameters are the same as those described in Step 3a, with these additional optional parameters:<br>• **ack**: Acknowledgment bit set.<br>• **established**: An established connection. A match occurs if the TCP datagram has the ACK or RST bits set.<br>• **fin**: Finished bit set; no more data from sender.<br>• **neq** { *port* \|**protocol**}: Matches only packets that are not on a given port number.<br>• **psh**—Push function bit set.<br>• **range** { *port* \|**protocol**}: Matches only packets in the port number range.<br>• **rst**: Reset bit set.<br>• **syn**: Synchronize bit set.<br>• **urg**: Urgent pointer bit set. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-ipv6-acl)# end` | Exits IPv6 access list configuration mode and returns to privileged EXEC mode. |
| **Step 7** | **show ipv6 access-list**<br>**Example:**<br>`Device# show ipv6 access-list` | Verifies that IPv6 ACLs are configured correctly. |

# Attaching an IPv6 ACL to an Interface

You can apply an ACL to outbound or inbound traffic on Layer 3 interfaces, or to inbound traffic on Layer 2 interfaces. You can also apply ACLs only to inbound management traffic on Layer 3 interfaces.

Follow these steps to control access to an interface.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Identifies a Layer 2 interface (for port ACLs) or Layer 3 interface (for router ACLs) on which to apply an access list, and enters interface configuration mode. |
| **Step 4** | **no switchport**<br>**Example:**<br>`Device(config-if)# no switchport` | Returns the interface to the routed-interface status and erases all further Layer 2 configuration. |
| **Step 5** | **ipv6 address** *ipv6-address*<br>**Example:**<br>`Device(config-if)# ipv6 address 2001:DB8::1` | Configures an IPv6 address on a Layer 3 interface (for router ACLs). |
| **Step 6** | **ipv6 traffic-filter** *access-list-name* {**in** \| **out**}<br>**Example:**<br>`Device(config-if)# ipv6 traffic-filter acl1 in` | Applies the access list to incoming or outgoing traffic on the interface. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-ipv6-acl)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring an IPv6 ACL in Template Mode

**Note** You can configure **ipv6 traffic-filter** command in the template configuration mode. You can configure the **source template** command only once to an interface.

Beginning in privileged EXEC mode, follow these steps to configure ACL in a template:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ipv6 access-list** {*list-name* \| **log-update threshold** \| **role-based** *list-name*}<br>**Example:**<br>`Device(config)# ipv6 access-list v6acl10` | Defines an IPv6 ACL name, and enters IPv6 access list configuration mode. |
| **Step 4** | **ipv6 access-list** {*list-name* \| **log-update threshold** \| **role-based** *list-name*}<br>**Example:**<br>`Device(config-ipv6-acl)#ipv6 access-list v6acl11` | Defines an IPv6 ACL name, and enters IPv6 access list configuration mode. |
| **Step 5** | **exit**<br>**Example:**<br>`Device(config-ipv6-acl)#exit` | Exits access-list configuration mode. |
| **Step 6** | **template**<br>**Example:**<br>`Device(config)# template test` | Creates a user template and enters template configuration mode. |
| **Step 7** | **ipv6 traffic-filter** {*access-list-number* \| *name*} {**in** \| **out**}<br>**Example:**<br>`Device(config-template)# ipv6 traffic-filter v6acl10 in` | Controls access to the specified interface.<br>Enter *access-list-number* to define the access list. The access list can be a number.<br>Enter *name* to define the access list. The access list can be a name.<br>Enter **in** to direct the access list in the incoming direction of the interface.<br>Enter **out** to direct the access list in the outgoing direction of the interface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **exit**<br>**Example:**<br>`Device(config-template)# exit` | Exits template configuration mode and returns to privileged EXEC mode. |
| **Step 9** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet1/0/1` | Identifies a specific interface for configuration, and enters interface configuration mode.<br>The interface can be a Layer 2 interface (port ACL), or a Layer 3 interface (router ACL). |
| **Step 10** | **ipv6 traffic-filter** {*access-list-number* \| *name*} {**in** \| **out**}<br>**Example:**<br>`Device(config-if)# ipv6 traffic-filter v6acl11 out` | Controls access to the specified interface.<br>Enter *access-list-number* to define the access list. The access list can be a number.<br>Enter *name* to define the access list. The access list can be a name.<br>Enter **in** to direct the access list in the incoming direction of the interface.<br>Enter **out** to direct the access list in the outgoing direction of the interface. |
| **Step 11** | **source template** *name*<br>**Example:**<br>`Device(config)# source template test` | Applies an interface template to a target. The accesss list *v6acl10* is the incoming access list that is configured. |
| **Step 12** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring a VLAN Map

To create a VLAN map and apply it to one or more VLANs, perform these steps:

### Before you begin

Create the IPv6 ACL that you want to apply to the VLAN.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **vlan access-map** *name* [**number**]<br>**Example:**<br>`Device(config)# vlan access-map map_1 20` | Creates a VLAN map, and enters VLAN access-map command mode<br>VLAN map can have a name or (optionally) a number. The number is the sequence number of the entry within the map.<br>When you create VLAN maps with the same name, numbers are assigned sequentially in increments of 10. When modifying or deleting maps, you can enter the number of the map entry that you want to modify or delete.<br>VLAN maps do not use the specific permit or deny keywords. To deny a packet by using VLAN maps, create an ACL that would match the packet, and set the action to drop. A permit in the ACL counts as a match. A deny in the ACL means no match. |
| **Step 4** | **match** {**ip** \| **ipv6** \| **mac**} **address** {*name* \| *number*} [*name* \| *number*]<br>**Example:**<br>`Device(config-access-map)# match ipv6 address ip_net` | Matches the packet against one or more access lists. Note that packets are only matched against access lists of the correct protocol type. IP packets are matched against IP access lists. Non-IP packets are only matched against named MAC access lists.<br>**Note**<br>If the VLAN map is configured with a match clause for a type of packet (IP or MAC) and the map action is drop, all packets that match the type are dropped. If the VLAN map has no match clause, and the configured action is drop, all IP and Layer 2 packets are dropped. |
| **Step 5** | Enter one of the following commands to specify an IP packet or a non-IP packet (with only a known MAC address) and to match the packet against one or more ACLs: | Sets the action for the map entry. |

| | Command or Action | Purpose |
|---|---|---|
| | • **action** { **forward**}<br>`Device(config-access-map)# action forward`<br>• **action** { **drop**}<br>`Device(config-access-map)# action drop` | |
| **Step 6** | **vlan filter** *mapname* **vlan-list** *list*<br>**Example:**<br>`Device(config)# vlan filter map 1 vlan-list 20-22` | Applies the VLAN map to one or more VLAN IDs.<br>The list can be a single VLAN ID (22), a consecutive list (10-22), or a string of VLAN IDs (12, 22, 30). Spaces around the comma and hyphen are optional. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Applying a VLAN Map to a VLAN

To apply a VLAN map to one or more VLANs, perform these steps.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **vlan filter** *mapname* **vlan-list** *list*<br>**Example:**<br>`Device(config)# vlan filter map 1 vlan-list 20-22` | Applies the VLAN map to one or more VLAN IDs.<br>The list can be a single VLAN ID (22), a consecutive list (10-22), or a string of VLAN IDs (12, 22, 30). Spaces around the comma and hyphen are optional. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Monitoring IPv6 ACLs

You can display information about all configured access lists, all IPv6 access lists, or a specific access list by using one or more of the privileged EXEC commands shown in the table below:

***Table 23: show ACL commands***

| Command | Purpose |
|---|---|
| **show access-lists** | Displays all access lists configured on the switch. |
| **show ipv6 access-list** [*access-list-name*] | Displays all configured IPv6 access lists or the access list specified by name. |
| **show vlan access-map** [*map-name*] | Displays VLAN access map configuration. |
| **show vlan filter** [**access-map** *access-map* \| **vlan** *vlan-id*] | Displays the mapping between VACLs and VLANs. |

# Configuration Examples for IPv6 ACL

This following sections display configuration examples for IPv6 ACL:

## Example: Creating an IPv6 ACL

This example configures the IPv6 access list named IPv6-ACL. The first deny entry in the list denies all packets that have a destination TCP port number greater than 5000. The second deny entry denies packets that have a source UDP port number less than 5000. The second deny also logs all matches to the console. The first permit entry in the list permits all ICMP packets. The second permit entry in the list permits all other traffic. The second permit entry is necessary because an implicit deny -all condition is at the end of each IPv6 access list.

**Note** Logging is supported only on Layer 3 interfaces.

```
Device> enable
Device(config)# ipv6 access-list IPv6_ACL
Device(config-ipv6-acl)# deny tcp any any gt 5000
Device (config-ipv6-acl)# deny ::/0 lt 5000 ::/0 log
Device(config-ipv6-acl)# permit icmp any any
Device(config-ipv6-acl)# permit any any
Device(config-ipv6-acl)# end
```

## Example: Displaying IPv6 ACLs

The following is a sample output from the **show access-lists** command. The output shows all access lists that are configured on the device.

```
Device# show access-lists

Extended IP access list hello
10 permit ip any any
IPv6 access list ipv6
permit ipv6 any any sequence 10
```

The following is a sample output from the **show ipv6 access-lists** command. The output shows only IPv6 access lists configured on the switch.

```
Device# show ipv6 access-list

IPv6 access list inbound
permit tcp any any eq bgp (8 matches) sequence 10
permit tcp any any eq telnet (15 matches) sequence 20
permit udp any any sequence 30
IPv6 access list outbound
deny udp any any sequence 10
deny tcp any any eq telnet sequence 20
```

# Example: Displaying VLAN Access Map Configuration

This is an example of the output from the **show vlan access-map** privileged EXEC command.

```
Device# show vlan access-map

Vlan access-map "m1"  10
  Match clauses:
    ipv6 address: ip2
  Action: drop
```

This is an example of the output from the show ipv6 access-lists privileged EXEC command. The output shows only IPv6 access lists configured on the switch.

```
Device# show ipv6 access-list

IPv6 access list inbound
permit tcp any any eq bgp (8 matches) sequence 10
permit tcp any any eq telnet (15 matches) sequence 20
permit udp any any sequence 30
IPv6 access list outbound
deny udp any any sequence 10
deny tcp any any eq telnet sequence 20
```

# Feature History for IPv6 ACLs

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | IPv6 ACLs | You can filter IPv6 traffic by creating IPv6 ACLs and applying them to interfaces similar to how you create and apply IPv4 named ACLs. You can also create and apply input router ACLs to filter Layer 3 management traffic.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, and C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | IPv6 ACLs | Support for this feature was introduced on the C9500-24Y4C, C9500-32C, C9500-32QC, and C9500-48Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Gibraltar 16.11.1 | IPv6 Downloadable ALCs | Support for IPv6 dACLs was introduced in this release |
| Cisco IOS XE Bengaluru 17.5.1 | ACL template support for IPv6 | Interface template allows you to configure multiple commands and associate it with an interface. The **ipv6 traffic-filter** command is used to apply an IPv6 access list under template mode of configuration. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 23

# Object Groups for ACLs

• Object Groups for ACLs, on page 401

# Object Groups for ACLs

The Object Groups for ACLs feature lets you classify users, devices, or protocols into groups and apply those groups to access control lists (ACLs) to create access control policies for those groups. This feature lets you use object groups instead of individual IP addresses, protocols, and ports, which are used in conventional ACLs. This feature allows multiple access control entries (ACEs), but now you can use each ACE to allow an entire group of users to access a group of servers or services or to deny them from doing so.

In large networks, the number of ACLs can be large (hundreds of lines) and difficult to configure and manage, especially if the ACLs frequently change. Object group-based ACLs are smaller, more readable, and easier to configure and manage than conventional ACLs, simplifying static and dynamic ACL deployments for large user access environments on Cisco IOS routers.

Cisco IOS Firewall benefits from object groups, because they simplify policy creation (for example, group A has access to group A services).

## Restrictions for Object Groups for ACLs

- You can use object groups only in extended named and numbered ACLs.
- Object group-based ACLs support only IPv4/IPv6 addresses.
- Object group-based ACLs support only Layer 3 interfaces(such as routed interfaces and VLAN interfaces) and sub-interfaces.
- Object group-based ACLs are not supported with IPsec.
- ACL statements using object groups will be ignored on packets that are sent to RP for processing.
- The number of object group-based ACEs supported in an ACL varies depending on platform, subject to TCAM availability.

## Information About Object Groups for ACLs

You can configure conventional ACEs and ACEs that refer to object groups in the same ACL.

You can use object group-based ACLs with quality of service (QoS) match criteria, Cisco IOS Firewall, Dynamic Host Configuration Protocol (DHCP), and any other features that use extended ACLs. In addition, you can use object group-based ACLs with multicast traffic.

When there are many inbound and outbound packets, using object group-based ACLs increases performance when compared to conventional ACLs. Also, in large configurations, this feature reduces the storage needed in NVRAM, because using object groups in ACEs means that you do not need to define an individual ACE for every address and protocol pairing.

## Object Groups

An object group can contain a single object (such as a single IP address, network, or subnet) or multiple objects (such as a combination of multiple IP addresses, networks, or subnets).

A typical access control entry (ACE) allows a group of users to have access only to a specific group of servers. In an object group-based access control list (ACL), you can create a single ACE that uses an object group name instead of creating many ACEs (which requires each ACE to have a different IP address). A similar object group (such as a protocol port group) can be extended to provide access only to a set of applications for a user group. ACEs can have object groups for the source only, destination only, none, or both.

You can use object groups to separate the ownership of the components of an ACE. For example, each department in an organization controls its group membership, and the administrator owns the ACE itself to control which departments can contact one another.

You can use object groups in features that use Cisco Policy Language (CPL) class maps.

This feature supports two types of object groups for grouping ACL parameters: network object groups and service object groups. Use these object groups to group IP addresses, protocols, protocol services (ports), and Internet Control Message Protocol (ICMP) types.

### Objects Allowed in Network Object Groups

A network object group is a group of any of the following objects:

- Any IP address—includes a range from 0.0.0.0 to 255.255.255.255 (This is specified using the **any** command.)
- Host IP addresses
- Hostnames
- Other network object groups
- Subnets
- Host IP addresses
- Network address of group members
- Nested object groups

### Objects Allowed in Service Object Groups

A service object group is a group of any of the following objects:

- Source and destination protocol ports (such as Telnet or Simple Network Management Protocol [SNMP])
- Internet Control Message Protocol (ICMP) types (such as echo, echo-reply, or host-unreachable)

- Top-level protocols (such as Encapsulating Security Payload [ESP], TCP, or UDP)
- Other service object groups

## ACLs Based on Object Groups

All features that use or reference conventional access control lists (ACLs) are compatible with object-group-based ACLs, and the feature interactions for conventional ACLs are the same with object-group-based ACLs. This feature extends the conventional ACLs to support object-group-based ACLs and also adds new keywords and the source and destination addresses and ports.

You can add, delete, or change objects in an object group membership list dynamically (without deleting and redefining the object group). Also, you can add, delete, or change objects in an object group membership list without redefining the ACL access control entry (ACE) that uses the object group. You can add objects to groups, delete them from groups, and then ensure that changes are correctly functioning within the object-group-based ACL without reapplying the ACL to the interface.

You can configure an object-group-based ACL multiple times with a source group only, a destination group only, or both source and destination groups.

You cannot delete an object group that is used within an ACL or a class-based policy language (CPL) policy.

# How to Configure Object Groups for ACLs

To configure object groups for ACLs, you first create one or more object groups. These can be any combination of network object groups (groups that contain objects such as, host addresses and network addresses) or service object groups (which use operators such as **lt**, **eq**, **gt**, **neq**, and **range** with port numbers). Then, you create access control entries (ACEs) that apply a policy (such as **permit** or **deny**) to those object groups.

## Creating a Network Object Group

A network object group that contains a single object (such as a single IP address, a hostname, another network object group, or a subnet) or multiple objects with a network object-group-based ACL to create access control policies for the objects.

Perform this task to create a network object group.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **object-group network** *object-group-name*<br>**Example:** | Defines the object group name and enters network object-group configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# object-group network my-network-object-group` | |
| **Step 4** | **description** *description-text*<br>**Example:**<br>`Device(config-network-group)# description test engineers` | (Optional) Specifies a description of the object group.<br>• You can use up to 200 characters. |
| **Step 5** | **host** {*host-address* \| *host-name*}<br>**Example:**<br>`Device(config-network-group)# host 209.165.200.237` | (Optional) Specifies the IP address or name of a host.<br>• If you specify a host address, you must use an IPv4 address. |
| **Step 6** | *network-address* {*/nn* \| *network-mask*}<br>**Example:**<br>`Device(config-network-group)# 209.165.200.225 255.255.255.224` | (Optional) Specifies a subnet object.<br>• You must specify an IPv4 address for the network address. The default network mask is 255.255.255.255. |
| **Step 7** | **group-object** *nested-object-group-name*<br>**Example:**<br>`Device(config-network-group)# group-object my-nested-object-group` | (Optional) Specifies a nested (child) object group to be included in the current (parent) object group.<br>• The type of child object group must match that of the parent (for example, if you are creating a network object group, you must specify another network object group as the child).<br>• You can use duplicated objects in an object group only via nesting of group objects. For example, if object 1 is in both group A and group B, you can define a group C that includes both A and B. However, you cannot include a group object that causes the group hierarchy to become circular (for example, you cannot include group A in group B and then also include group B in group A).<br>• You can use an unlimited number of levels of nested object groups (however, a maximum of two levels is recommended). |
| **Step 8** | Repeat the steps until you have specified objects on which you want to base your object group. | — |
| **Step 9** | **end**<br>**Example:**<br>`Device(config-network-group)# end` | Exits network object-group configuration mode and returns to privileged EXEC mode. |

## Creating a Service Object Group

Use a service object group to specify TCP and/or UDP ports or port ranges. When the service object group is associated with an access control list (ACL), this service object-group-based ACL can control access to ports.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **object-group service** *object-group-name*<br>**Example:**<br>`Device(config)# object-group service my-service-object-group` | Defines an object group name and enters service object-group configuration mode. |
| **Step 4** | **description** *description-text*<br>**Example:**<br>`Device(config-service-group)# description test engineers` | (Optional) Specifies a description of the object group.<br>• You can use up to 200 characters. |
| **Step 5** | *protocol*<br>**Example:**<br>`Device(config-service-group)# ahp` | (Optional) Specifies an IP protocol number or name. |
| **Step 6** | {**tcp** \| **udp** \| **tcp-udp**} [**source** {{[**eq**] \| **lt** \| **gt**} *port1* \| **range** *port1 port2*}] [{[**eq**] \| **lt** \| **gt**} *port1* \| **range** *port1 port2*]<br>**Example:**<br>`Device(config-service-group)# tcp-udp range 2000 2005` | (Optional) Specifies TCP, UDP, or both. |
| **Step 7** | **icmp** *icmp-type*<br>**Example:**<br>`Device(config-service-group)# icmp conversion-error` | (Optional) Specifies the decimal number or name of an Internet Control Message Protocol (ICMP) type. |
| **Step 8** | **group-object** *nested-object-group-name*<br>**Example:**<br>`Device(config-service-group)# group-object my-nested-object-group` | (Optional) Specifies a nested (child) object group to be included in the current (parent) object group. |

| | Command or Action | Purpose |
|---|---|---|
| | | • The type of child object group must match that of the parent (for example, if you are creating a network object group, you must specify another network object group as the child).<br>• You can use duplicated objects in an object group only via nesting of group objects. For example, if object 1 is in both group A and group B, you can define a group C that includes both A and B. However, you cannot include a group object that causes the group hierarchy to become circular (for example, you cannot include group A in group B and then also include group B in group A).<br>• You can use an unlimited number of levels of nested object groups (however, a maximum of two levels is recommended). |
| **Step 9** | Repeat the steps to specify the objects on which you want to base your object group. | — |
| **Step 10** | **end**<br>**Example:**<br>`Device(config-service-group)# end` | Exits service object-group configuration mode and returns to privileged EXEC mode. |

## Creating an Object-Group-Based ACL

When creating an object-group-based access control list (ACL), configure an ACL that references one or more object groups. As with conventional ACLs, you can associate the same access policy with one or more interfaces.

You can define multiple access control entries (ACEs) that reference object groups within the same object-group-based ACL. You can also reuse a specific object group in multiple ACEs.

Perform this task to create an object-group-based ACL.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip access-list extended** *access-list-name*<br>**Example:**<br>`Device(config)# ip access-list extended nomarketing` | Defines an extended IP access list using a name and enters extended access-list configuration mode. |
| **Step 4** | **remark** *remark*<br>**Example:**<br>`Device(config-ext-nacl)# remark protect server by denying access from the Marketing network` | (Optional) Adds a comment about the configured access list entry.<br>• A remark can precede or follow an access list entry.<br>• In this example, the remark reminds the network administrator that the subsequent entry denies the Marketing network access to the interface. |
| **Step 5** | **deny** *protocol source* [*source-wildcard*] *destination* [*destination-wildcard*] [**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**established**] [**log** \| **log-input**] [**time-range** *time-range-name*] [**fragments**]<br>**Example:**<br>`Device(config-ext-nacl)# deny ip 209.165.200.244 255.255.255.224 host 209.165.200.245 log`<br>`Example based on object-group:`<br>`Router(config)# object-group network my_network_object_group`<br>`Router(config-network-group)# 209.165.200.224 255.255.255.224`<br>`Router(config-network-group)# exit`<br>`Router(config)# object-group network my_other_network_object_group`<br>`Router(config-network-group)# host 209.165.200.245`<br>`Router(config-network-group)# exit`<br>`Router(config)# ip access-list extended nomarketing`<br>`Router(config-ext-nacl)# deny ip object-group my_network_object_group object-group my_other_network_object_group log` | (Optional) Denies any packet that matches all conditions specified in the statement.<br>• Optionally use the **object-group** *service-object-group-name* keyword and argument as a substitute for the *protocol.* argument<br>• Optionally use the **object-group** *source-network-object-group-name* keyword and argument as a substitute for the *source source-wildcard.* arguments<br>• Optionally use the **object-group** *destination-network-object-group-name* keyword and argument as a substitute for the *destination destination-wildcard.* arguments<br>• If the *source-wildcard* or *destination-wildcard*is omitted, a wildcard mask of 0.0.0.0 is assumed, which matches all bits of the source or destination address, respectively.<br>• Optionally use the **any** keyword as a substitute for the *source source-wildcard* or *destination destination-wildcard* to specify the address and wildcard of 0.0.0.0 255.255.255.255. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Optionally use the **host** *source* keyword and argument to indicate a source and source wildcard of *source* 0.0.0.0 or the **host** *destination* keyword and argument to indicate a destination and destination wildcard of *destination* 0.0.0.0.<br>• In this example, packets from all sources are denied access to the destination network 209.165.200.244. Logging messages about packets permitted or denied by the access list are sent to the facility configured by the **logging facility** command (for example, console, terminal, or syslog). That is, any packet that matches the access list will cause an informational logging message about the packet to be sent to the configured facility. The level of messages logged to the console is controlled by the **logging console** command. |
| **Step 6** | **remark** *remark*<br>**Example:**<br>`Device(config-ext-nacl)# remark allow TCP from any source to any destination` | (Optional) Adds a comment about the configured access list entry.<br>• A remark can precede or follow an access list entry. |
| **Step 7** | **permit** *protocol source* [*source-wildcard*] *destination* [*destination-wildcard*] [**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**established**] [**log** \| **log-input**] [**time-range** *time-range-name*] [**fragments**]<br>**Example:**<br>`Device(config-ext-nacl)# permit tcp any any` | Permits any packet that matches all conditions specified in the statement.<br>• Every access list needs at least one permit statement.<br>• Optionally use the **object-group** *service-object-group-name* keyword and argument as a substitute for the *protocol.*<br>• Optionally use the **object-group** *source-network-object-group-name* keyword and argument as a substitute for the *source source-wildcard.*<br>• Optionally use the **object-group** *destination-network-object-group-name* keyword and argument as a substitute for the *destination destination-wildcard.*<br>• If *source-wildcard* or *destination-wildcard* is omitted, a wildcard mask of 0.0.0.0 is assumed, which matches on all bits of the source or destination address, respectively. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Optionally use the **any**keyword as a substitute for the *source source-wildcard* or *destination destination-wildcard* to specify the address and wildcard of 0.0.0.0 255.255.255.255.<br>• In this example, TCP packets are allowed from any source to any destination.<br>• Use the **log-input** keyword to include input interface, source MAC address, or virtual circuit in the logging output. |
| **Step 8** | Repeat the steps to specify the fields and values on which you want to base your access list. | Remember that all sources not specifically permitted are denied by an implicit **deny** statement at the end of the access list. |
| **Step 9** | **end**<br>**Example:**<br>`Device(config-ext-nacl)# end` | Exits extended access-list configuration mode and returns to privileged EXEC mode. |

## Applying an Object Group-Based ACL to an Interface

Use the **ip access-group** command to apply an object group-based ACL to an interface. An object group-based access control list (ACL) can be used to control traffic on the interface it is applied to.

Perform this task to apply an object group-based ACL to an interface.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface vlan 100` | Specifies the interface and enters interface configuration mode. |
| **Step 4** | **ip access-group** {*access-list-name* \| *access-list-number*} {**in** \| **out**}<br>**Example:** | Applies the ACL to the interface and specifies whether to filter inbound or outbound packets. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-if)# ip access-group my-ogacl-policy in` | |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

## Verifying Object Groups for ACLs

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **show object-group** [*object-group-name*]<br>**Example:**<br>`Device# show object-group my-object-group` | Displays the configuration in the named or numbered object group (or in all object groups if no name is entered). |
| **Step 3** | **show ip access-list** [*access-list-name*]<br>**Example:**<br>`Device# show ip access-list my-ogacl-policy` | Displays the contents of the named or numbered access list or object group-based ACL (or for all access lists and object group-based ACLs if no name is entered). |

# Configuration Examples for Object Groups for ACLs

## Example: Creating a Network Object Group

The following example shows how to create a network object group named my-network-object-group, which contains two hosts and a subnet as objects:

```
Device> enable
Device# configure terminal
Device(config)# object-group network my-network-object-group
Device(config-network-group)# description test engineers
Device(config-network-group)# host 209.165.200.237
Device(config-network-group)# host 209.165.200.238

Device(config-network-group)# 209.165.200.241 255.255.255.224
Device(config-network-group)# end
```

The following example shows how to create a network object group named my-company-network, which contains two hosts, a subnet, and an existing object group (child) named my-nested-object-group as objects:

```
Device> enable
Device# configure terminal
Device(config)# object-group network my-company-network
Device(config-network-group)# host host1
```

```
Device(config-network-group)# host 209.165.200.242
Device(config-network-group)# 209.165.200.225 255.255.255.224
Device(config-network-group)# group-object my-nested-object-group
Device(config-network-group)# end
```

## Example: Creating a Service Object Group

The following example shows how to create a service object group named my-service-object-group, which contains several ICMP, TCP, UDP, and TCP-UDP protocols and an existing object group named my-nested-object-group as objects:

```
Device> enable
Device# configure terminal
Device(config)# object-group service my-service-object-group
Device(config-service-group)# icmp echo
Device(config-service-group)# tcp smtp
Device(config-service-group)# tcp telnet
Device(config-service-group)# tcp source range 1 65535 telnet
Device(config-service-group)# tcp source 2000 ftp
Device(config-service-group)# udp domain
Device(config-service-group)# tcp-udp range 2000 2005
Device(config-service-group)# group-object my-nested-object-group
Device(config-service-group)# end
```

## Example: Creating an Object Group-Based ACL

The following example shows how to create an object-group-based ACL that permits packets from the users in my-network-object-group if the protocol ports match the ports specified in my-service-object-group:

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended my-ogacl-policy
Device(config-ext-nacl)# permit object-group my-service-object-group object-group
my-network-object-group any
Device(config-ext-nacl)# deny tcp any any
Device(config-ext-nacl)# end
```

## Applying an Object Group-Based ACL to an Interface

Use the **ip access-group** command to apply an object group-based ACL to an interface. An object group-based access control list (ACL) can be used to control traffic on the interface it is applied to.

Perform this task to apply an object group-based ACL to an interface.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface vlan 100` | Specifies the interface and enters interface configuration mode. |
| **Step 4** | **ip access-group** {*access-list-name* \| *access-list-number*} {**in** \| **out**}<br>**Example:**<br>`Device(config-if)# ip access-group my-ogacl-policy in` | Applies the ACL to the interface and specifies whether to filter inbound or outbound packets. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

## Example: Verifying Object Groups for ACLs

The following example shows how to display all object groups:

```
Device# show object-group

Network object group auth-proxy-acl-deny-dest
 host 209.165.200.235
Service object group auth-proxy-acl-deny-services
 tcp eq www
 tcp eq 443
Network object group auth-proxy-acl-permit-dest
 209.165.200.226 255.255.255.224
 209.165.200.227 255.255.255.224
 209.165.200.228 255.255.255.224
 209.165.200.229 255.255.255.224
 209.165.200.246 255.255.255.224
 209.165.200.230 255.255.255.224
 209.165.200.231 255.255.255.224
 209.165.200.232 255.255.255.224
 209.165.200.233 255.255.255.224
 209.165.200.234 255.255.255.224
Service object group auth-proxy-acl-permit-services
 tcp eq www
 tcp eq 443
```

The following example shows how to display information about specific object-group-based ACLs:

```
Device# show ip access-list my-ogacl-policy

Extended IP access list my-ogacl-policy
10 permit object-group eng_service any any
```

# Additional References for Object Groups for ACLs

### Related Documents

| Related Topic | Document Title |
|---|---|
| Security commands | • Cisco IOS Security Command Reference: Commands A to C<br>• Cisco IOS Security Command Reference: Commands D to L<br>• Cisco IOS Security Command Reference: Commands M to R<br>• Cisco IOS Security Command Reference: Commands S to Z |
| ACL configuration guide | *Security Configuration Guide: Access Control Lists* |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password. | http://www.cisco.com/cisco/web/support/index.html |

# Feature History for Object Groups for ACLs

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Gibraltar 16.12.1 | Object Groups for ACLs | The Object Groups for ACLs feature lets you classify users, devices, or protocols into groups and apply them to access control lists (ACLs) to create access control policies for those groups. This feature lets you use object groups instead of individual IP addresses, protocols, and ports, which are used in conventional ACLs. This feature allows multiple access control entries (ACEs), but now you can use each ACE to allow an entire group of users to access a group of servers or services or to deny them from doing so. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 24

# Configuring IP Source Guard


- Information About IP Source Guard, on page 415
- How to Configure IP Source Guard, on page 417
- Monitoring IP Source Guard, on page 419
- Feature History for IP Source Guard, on page 420


# Information About IP Source Guard

## IP Source Guard

You can use IP source guard (IPSG) to prevent traffic attacks if a host tries to use the IP address of its neighbor and you can enable IP source guard when DHCP snooping is enabled on an untrusted interface.

After IPSG is enabled on an interface, the switch blocks all IP traffic received on the interface except for DHCP packets allowed by DHCP snooping.

The switch uses a source IP lookup table in hardware to bind IP addresses to ports. For IP and MAC filtering, a combination of source IP and source MAC lookups are used. IP traffic with a source IP address in the binding table is allowed, all other traffic is denied.

The IP source binding table has bindings that are learned by DHCP snooping or are manually configured (static IP source bindings). An entry in this table has an IP address, its associated MAC address, and its associated VLAN number. The switch uses the IP source binding table only when IP source guard is enabled.

IPSG is supported only on Layer 2 ports, including access and trunk ports. You can configure IPSG with source IP address filtering or with source IP and MAC address filtering.

## IP Source Guard for Static Hosts

**Note** Do not use IPSGfor static hosts on uplink ports or trunk ports.

IPSG for static hosts extends the IPSG capability to non-DHCP and static environments. The previous IPSG used the entries created by DHCP snooping to validate the hosts connected to a switch. Any traffic received from a host without a valid DHCP binding entry is dropped. This security feature restricts IP traffic on nonrouted Layer 2 interfaces. It filters traffic based on the DHCP snooping binding database and on manually

configured IP source bindings. The previous version of IPSG required a DHCP environment for IPSG to work.

IPSG for static hosts allows IPSG to work without DHCP. IPSG for static hosts relies on IP device tracking-table entries to install port ACLs. The switch creates static entries based on ARP requests or other IP packets to maintain the list of valid hosts for a given port. You can also specify the number of hosts allowed to send traffic to a given port. This is equivalent to port security at Layer 3.

IPSG for static hosts also supports dynamic hosts. If a dynamic host receives a DHCP-assigned IP address that is available in the IP DHCP snooping table, the same entry is learned by the IP device tracking table. In a stacked environment, when the active switch failover occurs, the IP source guard entries for static hosts attached to member ports are retained. When you enter the **show device-tracking database** EXEC command, the IP device tracking table displays the entries as ACTIVE.

**Note** Some IP hosts with multiple network interfaces can inject some invalid packets into a network interface. The invalid packets contain the IP or MAC address for another network interface of the host as the source address. The invalid packets can cause IPSG for static hosts to connect to the host, to learn the invalid IP or MAC address bindings, and to reject the valid bindings. Consult the vender of the corresponding operating system and the network interface to prevent the host from injecting invalid packets.

IPSG for static hosts initially learns IP or MAC bindings dynamically through an ACL-based snooping mechanism. IP or MAC bindings are learned from static hosts by ARP and IP packets. They are stored in the device tracking database. When the number of IP addresses that have been dynamically learned or statically configured on a given port reaches a maximum, the hardware drops any packet with a new IP address. To resolve hosts that have moved or gone away for any reason, IPSG for static hosts leverages IP device tracking to age out dynamically learned IP address bindings. This feature can be used with DHCP snooping. Multiple bindings are established on a port that is connected to both DHCP and static hosts. For example, bindings are stored in both the device tracking database as well as in the DHCP snooping binding database.

# IP Source Guard Configuration Guidelines

- You can configure static IP bindings only on nonrouted ports. If you enter the **ip source binding** *mac-address* **vlan** *vlan-id ip-address* **interface** *interface-id* global configuration command on a routed interface, this error message appears:

  ```
  Static IP source binding can only be configured on switch port.
  ```

- When IP source guard with source IP filtering is enabled on an interface, DHCP snooping must be enabled on the access VLAN for that interface.
- If you are enabling IP source guard on a trunk interface with multiple VLANs and DHCP snooping is enabled on all the VLANs, the source IP address filter is applied on all the VLANs.

**Note** If IP source guard is enabled and you enable or disable DHCP snooping on a VLAN on the trunk interface, the switch might not properly filter traffic.

- You can enable this feature when 802.1x port-based authentication is enabled.

- When you configure IP source guard smart logging, packets with a source address other than the specified address or an address learned by DHCP are denied, and the packet contents are sent to a NetFlow collector. If you configure this feature, make sure that smart logging is globally enabled.
- In a switch stack, if IP source guard is configured on a stack member interface and you remove the the configuration of that switch by entering the **no switch** *stack-member-number* **provision** global configuration command, the interface static bindings are removed from the binding table, but they are not removed from the running configuration. If you again provision the switch by entering the **switch** *stack-member-number* **provision** command, the binding is restored.

  To remove the binding from the running configuration, you must disable IP source guard before entering the **no switch provision** command. The configuration is also removed if the switch reloads while the interface is removed from the binding table.

# How to Configure IP Source Guard

## Enabling IP Source Guard

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Specifies the interface to be configured, and enters interface configuration mode. |
| **Step 4** | **ip verify source** [**mac-check** ]<br>**Example:**<br>`Device(config-if)# ip verify source` | Enables IP source guard with source IP address filtering.<br>(Optional) **mac-check**: Enables IP Source Guard with source IP address and MAC address filtering. |
| **Step 5** | **exit**<br>**Example:**<br>`Device(config-if)# exit` | Exits interface configuration mode and returns to global configuration mode. |
| **Step 6** | **ip source binding** *mac-address* **vlan** *vlan-id* *ip-address* **interface** *interface-id* | Adds a static IP source binding. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device(config)# ip source binding 0100.0230.0002 vlan 11 10.0.0.4 interface gigabitethernet1/0/1` | Enter this command for each static binding. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and reutrns to privileged EXEC mode. |

# Configuring IP Source Guard for Static Hosts on a Layer 2 Access Port

You must configure the **ip device tracking maximum** *limit-number* interface configuration command globally for IPSG for static hosts to work. If you only configure this command on a port without enabling IP device tracking globally or by setting an IP device tracking maximum on that interface, IPSG with static hosts rejects all the IP traffic from that interface.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip device tracking**<br>**Example:**<br>`Device(config)# ip device tracking` | Turns on the IP host table, and globally enables IP device tracking. |
| **Step 4** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Enters interface configuration mode. |
| **Step 5** | **switchport mode access**<br>**Example:**<br>`Device(config-if)# switchport mode access` | Configures a port as access. |
| **Step 6** | **switchport access vlan** *vlan-id*<br>**Example:**<br>`Device(config-if)# switchport access vlan 10` | Configures the VLAN for this port. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **ip verify source**[**tracking**] [**mac-check** ]<br>**Example:**<br>`Device(config-if)# ip verify source tracking mac-check` | Enables IP source guard with source IP address filtering.<br>(Optional) **tracking**: Enables IP source guard for static hosts.<br>(Optional) **mac-check**: Enables MAC address filtering.<br>The command **ip verify source tracking mac-check**enables IP source guard for static hosts with MAC address filtering. |
| **Step 8** | **ip device tracking maximum** *number*<br>**Example:**<br>`Device(config-if)# ip device tracking maximum 8` | Establishes a maximum limit for the number of static IPs that the IP device tracking table allows on the port. The range is 1to 10. The maximum number is 10.<br>**Note**<br>You must configure the **ip device tracking maximum** *limit-number* interface configuration command. |
| **Step 9** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Monitoring IP Source Guard

*Table 24: Privileged EXEC show Commands*

| Command | Purpose |
|---|---|
| **show ip verify source** [ **interface** *interface-id* ] | Displays the IP source guard configuration on the switch or on a specific interface. |
| **show ip device tracking** { **all** \| **interface** *interface-id* \| **ip** *ip-address* \| **mac** *mac-address*} | Displays information about the entries in the IP device tracking table. |

*Table 25: Interface Configuration Commands*

| Command | Purpose |
|---|---|
| **ip verify source tracking** | Verifies the data source. |

# Feature History for IP Source Guard

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | IP Source Guard | You can use IP source guard to prevent traffic attacks if a host tries to use the IP address of its neighbor and you can enable IP source guard when DHCP snooping is enabled on an untrusted interface.<br><br>Support for this feature was introduced on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | IP Source Guard | Support for this feature was introduced on the C9500-24Y4C, C9500-32C, C9500-32QC, and C9500-48Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

CHAPTER 25

# Configuring Dynamic ARP Inspection


- Restrictions for Dynamic ARP Inspection, on page 421
- Information About Dynamic ARP Inspection, on page 422
- How to Configure Dynamic ARP Inspection, on page 426
- Monitoring DAI, on page 434
- Verifying the DAI Configuration, on page 434
- Feature History for Dynamic ARP Inspection, on page 434


## Restrictions for Dynamic ARP Inspection

This section lists the restrictions and guidelines for configuring Dynamic Address Resolution Protocol (ARP) Inspection on the switch.

- Dynamic ARP inspection is an ingress security feature; it does not perform any egress checking.
- Dynamic ARP inspection is not effective for hosts connected to switches that do not support dynamic ARP inspection or that do not have this feature enabled. Because man-in-the-middle attacks are limited to a single Layer 2 broadcast domain, separate the domain with dynamic ARP inspection checks from the one with no checking. This action secures the ARP caches of hosts in the domain enabled for dynamic ARP inspection.
- Dynamic ARP inspection depends on the entries in the DHCP snooping binding database to verify IP-to-MAC address bindings in incoming ARP requests and ARP responses. Make sure to enable DHCP snooping to permit ARP packets that have dynamically assigned IP addresses.

  When DHCP snooping is disabled or in non-DHCP environments, use ARP ACLs to permit or to deny packets.
- Dynamic ARP inspection is supported on access ports, trunk ports, and EtherChannel ports.

**Note** Do not enable Dynamic ARP inspection on RSPAN VLANs. If Dynamic ARP inspection is enabled on RSPAN VLANs, Dynamic ARP inspection packets might not reach the RSPAN destination port.

- A physical port can join an EtherChannel port channel only when the trust state of the physical port and the channel port match. Otherwise, the physical port remains suspended in the port channel. A port

channel inherits its trust state from the first physical port that joins the channel. Consequently, the trust state of the first physical port need not match the trust state of the channel.

Conversely, when you change the trust state on the port channel, the switch configures a new trust state on all the physical ports that comprise the channel.

- The rate limit is calculated separately on each switch in a switch stack. For a cross-stack EtherChannel, this means that the actual rate limit might be higher than the configured value. For example, if you set the rate limit to 30 pps on an EtherChannel that has one port on switch 1 and one port on switch 2, each port can receive packets at 29 pps without causing the EtherChannel to become error-disabled.
- The operating rate for the port channel is cumulative across all the physical ports within the channel. For example, if you configure the port channel with an ARP rate-limit of 400 pps, all the interfaces combined on the channel receive an aggregate 400 pps. The rate of incoming ARP packets on EtherChannel ports is equal to the sum of the incoming rate of packets from all the channel members. Configure the rate limit for EtherChannel ports only after examining the rate of incoming ARP packets on the channel-port members.

  The rate of incoming packets on a physical port is checked against the port-channel configuration rather than the physical-ports configuration. The rate-limit configuration on a port channel is independent of the configuration on its physical ports.

  If the EtherChannel receives more ARP packets than the configured rate, the channel (including all physical ports) is placed in the error-disabled state.
- Make sure to limit the rate of ARP packets on incoming trunk ports. Configure trunk ports with higher rates to reflect their aggregation and to handle packets across multiple dynamic ARP inspection-enabled VLANs. You also can use the **ip arp inspection limit none** interface configuration command to make the rate unlimited. A high rate-limit on one VLAN can cause a denial-of-service attack to other VLANs when the software places the port in the error-disabled state.
- When you enable dynamic ARP inspection on the switch, policers that were configured to police ARP traffic are no longer effective. The result is that all ARP traffic is sent to the CPU.

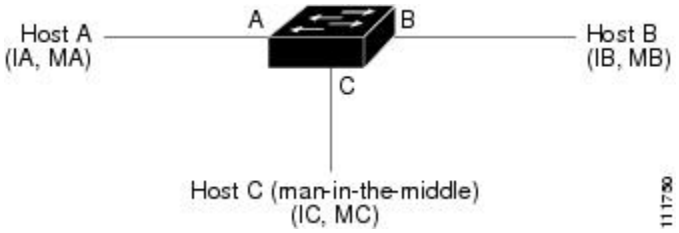
# Information About Dynamic ARP Inspection

## Understanding Dynamic ARP Inspection

ARP provides IP communication within a Layer 2 broadcast domain by mapping an IP address to a MAC address. For example, Host B wants to send information to Host A but does not have the MAC address of Host A in its ARP cache. Host B generates a broadcast message for all hosts within the broadcast domain to obtain the MAC address associated with the IP address of Host A. All hosts within the broadcast domain receive the ARP request, and Host A responds with its MAC address. However,because ARP allows a gratuitous reply from a host even if an ARP request was not received, an ARP spoofing attack and the poisoning of ARP caches can occur. After the attack, all traffic from the device under attack flows through the attacker's computer and then to the router, switch, or host.

A malicious user can attack hosts, switches, and routers connected to your Layer 2 network by poisoning the ARP caches of systems connected to the subnet and by intercepting traffic intended for other hosts on the subnet. Figure 26-1 shows an example of ARP cache poisoning.

*Figure 21: ARP Cache Poisoning*




Hosts A, B, and C are connected to the switch on interfaces A, B and C, all of which are on the same subnet. Their IP and MAC addresses are shown in parentheses; for example, Host A uses IP address IA and MAC address MA. When Host A needs to communicate to Host B at the IP layer, it broadcasts an ARP request for the MAC address associated with IP address IB. When the switch and Host B receive the ARP request, they populate their ARP caches with an ARP binding for a host with the IP address IA and a MAC address MA; for example, IP address IA is bound to MAC address MA. When Host B responds, the switch and Host A populate their ARP caches with a binding for a host with the IP address IB and the MAC address MB.

Host C can poison the ARP caches of the switch, Host A, and Host B by broadcasting forged ARP responses with bindings for a host with an IP address of IA (or IB) and a MAC address of MC. Hosts with poisoned ARP caches use the MAC address MC as the destination MAC address for traffic intended for IA or IB. This means that Host C intercepts that traffic. Because Host C knows the true MAC addresses associated with IA and IB, it can forward the intercepted traffic to those hosts by using the correct MAC address as the destination. Host C has inserted itself into the traffic stream from Host A to Host B, the classic *man-in-the middle*attack.

Dynamic ARP inspection is a security feature that validates ARP packets in a network. It intercepts, logs,and discards ARP packets with invalid IP-to-MAC address bindings. This capability protects the network from certain man-in-the-middle attacks.

Dynamic ARP inspection ensures that only valid ARP requests and responses are relayed. The switch performs these activities:

- Intercepts all ARP requests and responses on untrusted ports.
- Verifies that each of these intercepted packets has a valid IP-to-MAC address binding before updating the local ARP cache or before forwarding the packet to the appropriate destination.
- Drops invalid ARP packets.

Dynamic ARP inspection determines the validity of an ARP packet based on valid IP-to-MAC address bindings stored in a trusted database, the DHCP snooping binding database. This database is built by DHCP snooping if DHCP snooping is enabled on the VLANs and on the switch. If the ARP packet is received on a trusted interface, the switch forwards the packet without any checks. On untrusted interfaces, the switch forwards the packet only if it is valid.

In non-DHCP environments, dynamic ARP inspection can validate ARP packets against user-configured ARP access control lists (ACLs) for hosts with statically configured IP addresses. You define an ARP ACL by using the **arp access-list** *acl-name* global configuration command.

You can configure dynamic ARP inspection to drop ARP packets when the IP addresses in the packets are invalid or when the MAC addresses in the body of the ARP packets do not match the addresses specified in the Ethernet header. Use the **ip arp inspection validate** {[**src-mac**] [**dst-mac**] [**ip**]} global configuration command.

## Interface Trust States and Network Security

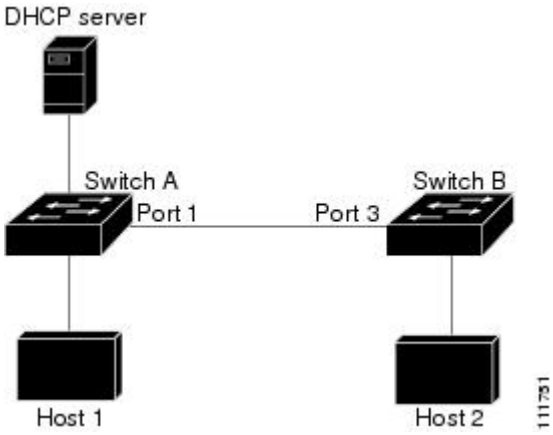
Dynamic ARP inspection associates a trust state with each interface on the switch. Packets arriving on trusted interfaces bypass all dynamic ARP inspection validation checks, and those arriving on untrusted interfaces undergo the dynamic ARP inspection validation process.

In a typical network configuration, you configure all switch ports connected to host ports as untrusted and configure all switch ports connected to switches as trusted. With this configuration, all ARP packets entering the network from a given switch bypass the security check. No other validation is needed at any other place in the VLAN or in the network. You configure the trust setting by using theip arp inspection trust interface configuration command.

**Caution** Use the trust state configuration carefully. Configuring interfaces as untrusted when they should betrusted can result in a loss of connectivity.

In the following figure, assume that both Switch A and Switch B are running dynamic ARP inspection on the VLAN that includes Host 1 and Host 2. If Host 1 and Host 2 acquire their IP addresses from the DHCP server connected to Switch A, only Switch A binds the IP-to-MAC address of Host 1. Therefore, if the interface between Switch A and Switch B is untrusted, the ARP packets from Host 1 are dropped by Switch B. Connectivity between Host 1 and Host 2 is lost.

***Figure 22: ARP Packet Validation on a VLAN Enabled for Dynamic ARP Inspection***




Configuring interfaces to be trusted when they are actually untrusted leaves a security hole in the network. If Switch A is not running dynamic ARP inspection, Host 1 can easily poison the ARP cache of Switch B (and Host 2, if the link between the switches is configured as trusted). This condition can occur even though Switch B is running dynamic ARP inspection.

Dynamic ARP inspection ensures that hosts (on untrusted interfaces) connected to a switch running dynamic ARP inspection do not poison the ARP caches of other hosts in the network. However, dynamic ARP inspection does not prevent hosts in other portions of the network from poisoning the caches of the hosts that are connected to a switch running dynamic ARP inspection.

In cases in which some switches in a VLAN run dynamic ARP inspection and other switches do not, configure the interfaces connecting such switches as untrusted. However, to validate the bindings of packets from nondynamic ARP inspection switches, configure the switch running dynamic ARP inspection with ARP ACLs. When you cannot determine such bindings, at Layer 3, isolate switches running dynamic ARP inspection from switches not running dynamic ARP inspection switches.

**Note** Depending on the setup of the DHCP server and the network, it might not be possible to validate a given ARP packet on all switches in the VLAN.

## Rate Limiting of ARP Packets

The switch CPU performs dynamic ARP inspection validation checks; therefore, the number of incoming ARP packets is rate-limited to prevent a denial-of-service attack. By default, the rate for untrusted interfaces is 15 packets per second (pps). Trusted interfaces are not rate-limited. You can change this setting by using the **ip arp inspection limit** interface configuration command.

When the rate of incoming ARP packets exceeds the configured limit, the switch places the port in the error-disabled state. The port remains in that state until you intervene. You can use the **errdisable recovery** global configuration command to enable error disable recovery so that ports automatically emerge from this state after a specified timeout period.

**Note** The rate limit for an EtherChannel is applied separately to each switch in a stack. For example, if a limit of 20 pps is configured on the EtherChannel, each switch with ports in the EtherChannel can carry up to 20 pps. If any switch exceeds the limit, the entire EtherChannel is placed into the error-disabled state.

## Relative Priority of ARP ACLs and DHCP Snooping Entries

Dynamic ARP inspection uses the DHCP snooping binding database for the list of valid IP-to-MAC address bindings.

ARP ACLs take precedence over entries in the DHCP snooping binding database. The switch uses ACLs only if you configure them by using the **ip arp inspection filter vlan** global configuration command. The switch first compares ARP packets to user-configured ARP ACLs. If the ARP ACL denies the ARP packet, the switch also denies the packet even if a valid binding exists in the database populated by DHCP snooping.

## Logging of Dropped Packets

When the switch drops a packet, it places an entry in the log buffer and then generates system messages on a rate-controlled basis. After the message is generated, the switch clears the entry from the log buffer. Each log entry contains flow information, such as the receiving VLAN, the port number, the source and destination IP addresses, and the source and destination MAC addresses.

You use the **ip arp inspection log-buffer** global configuration command to configure the number of entries in the buffer and the number of entries needed in the specified interval to generate system messages. You specify the type of packets that are logged by using the **ip arp inspection vlan logging** global configuration command.

# Default Dynamic ARP Inspection Configuration

| Feature | Default Settings |
|---|---|
| Dynamic ARP inspection | Disabled on all VLANs. |
| Interface trust state | All interfaces are untrusted. |

| Feature | Default Settings |
|---|---|
| Rate limit of incoming ARP packets | The rate is 15 pps on untrusted interfaces, assuming that the network is a switched network with a host connecting to as many as 15 new hosts per second.<br>The rate is unlimited on all trusted interfaces.<br>The burst interval is 1 second. |
| ARP ACLs for non-DHCP environments | No ARP ACLs are defined. |
| Validation checks | No checks are performed. |
| Log buffer | When dynamic ARP inspection is enabled, all denied or dropped ARP packets are logged.<br>The number of entries in the log is 32.<br>The number of system messages is limited to 5 per second.<br>The logging-rate interval is 1 second. |
| Per-VLAN logging | All denied or dropped ARP packets are logged. |

## Relative Priority of ARP ACLs and DHCP Snooping Entries

Dynamic ARP inspection uses the DHCP snooping binding database for the list of valid IP-to-MAC address bindings.

ARP ACLs take precedence over entries in the DHCP snooping binding database. The switch uses ACLs only if you configure them by using the ip arp inspection filter vlan global configuration command. The switch first compares ARP packets to user-configured ARP ACLs. If the ARP ACL denies the ARP packet, the switch also denies the packet even if a valid binding exists in the database populated by DHCP snooping.

# How to Configure Dynamic ARP Inspection

## Configuring ARP ACLs for Non-DHCP Environments

This procedure shows how to configure dynamic ARP inspection when Switch B shown in Figure 2 does not support dynamic ARP inspection or DHCP snooping.

If you configure port 1 on Switch A as trusted, a security hole is created because both Switch A and Host 1 could be attacked by either Switch B or Host 2. To prevent this possibility, you must configure port 1 on Switch A as untrusted. To permit ARP packets from Host 2, you must set up an ARP ACL and apply it to VLAN 1. If the IP address of Host 2 is not static (it is impossible to apply the ACL configuration on Switch A) you must separate Switch A from Switch B at Layer 3 and use a router to route packets between them.

Follow these steps to configure an ARP ACL on Switch A. This procedure is required in non-DHCP environments.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **arp access-list** *acl-name*<br>**Example:**<br>`Device(config)# arp access-list arpacl22` | Defines an ARP ACL, and enters ARP access-list configuration mode. By default, no ARP access lists are defined.<br>**Note**<br>At the end of the ARP access list, there is an implicit **deny ip any mac any** command. |
| **Step 4** | **permit ip host** *sender-ip* **mac host** *sender-mac*<br>**Example:**<br>`Device(config-arp-nacl))# permit ip host 10.2.2.2 mac host 0018.bad8.3fbd` | Permits ARP packets from the specified host (Host 2).<br>• For*sender-ip*, enter the IP address of Host 2.<br>• For *sender-mac*, enter the MAC address of Host 2. |
| **Step 5** | **exit**<br>**Example:**<br>`Device(config-arp-nacl)# exit` | Exits ARP access-list configuration mode and returns to global configuration mode. |
| **Step 6** | **ip arp inspection filter** *arp-acl-name* **vlan** *vlan-range* [**static**]<br>**Example:**<br>`Device(config)# ip arp inspection filter arpacl22 vlan 1-2` | Applies ARP ACL to the VLAN. By default, no defined ARP ACLs are applied to any VLAN.<br>• For *arp-acl-name*, specify the name of the ACL created in Step 2.<br>• For *vlan-range*, specify the VLAN that the switches and hosts are in. You can specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs separated by a comma. The range is 1 to 4094.<br>• (Optional) Specify **static** to treat implicit denies in the ARP ACL as explicit denies and to drop packets that do not match any |

| | Command or Action | Purpose |
|---|---|---|
| | | previous clauses in the ACL. DHCP bindings are not used.<br>If you do not specify this keyword, it means that there is no explicit deny in the ACL that denies the packet, and DHCP bindings determine whether a packet is permitted or denied if the packet does not match any clauses in the ACL.<br>ARP packets containing only IP-to-MAC address bindings are compared against the ACL. Packets are permitted only if the access list permits them. |
| **Step 7** | **interface** *interface-type interface-number*<br>**Example:**<br>`Device(config)# interface gigabitethernt 0/1/1` | Specifies Switch A interface that is connected to Switch B, and enters interface configuration mode. |
| **Step 8** | **no ip arp inspection trust**<br>**Example:**<br>`Device(config-if)# no ip arp inspection trust` | Configures Switch A interface that is connected to Switch B as untrusted.<br>By default, all interfaces are untrusted.<br>For untrusted interfaces, the switch intercepts all ARP requests and responses. It verifies that the intercepted packets have valid IP-to-MAC address bindings before updating the local cache and before forwarding the packet to the appropriate destination. The switch drops invalid packets and logs them in the log buffer according to the logging configuration specified with the **ip arp inspection vlan logging** global configuration command. |
| **Step 9** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 10** | **show arp access-list** *acl-name*<br>**Example:**<br>`Device# show arp access-list arpacl22` | Displays information about the named ACLs. |
| **Step 11** | **show ip arp inspection vlan** *vlan-range*<br>**Example:**<br>`Device# show ip arp inspection vlan 1-2` | Displays the statistics for the selected range of VLANs. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 12** | **show ip arp inspection interfaces**<br>**Example:**<br>`Device# show ip arp inspection interfaces` | Displays the trust state and the rate limit of ARP packets for the provided interface. |

# Configuring Dynamic ARP Inspection in DHCP Environments

### Before you begin

This procedure shows how to configure dynamic ARP inspection when two switches support this feature. Host 1 is connected to Switch A, and Host 2 is connected to Switch B. Both switches are running dynamic ARP inspection on VLAN 1 where the hosts are located. A DHCP server is connected to Switch A. Both hosts acquire their IP addresses from the same DHCP server. Therefore, Switch A has the bindings for Host 1 and Host 2, and Switch B has the binding for Host 2.

**Note** Dynamic ARP inspection depends on the entries in the DHCP snooping binding database to verify IP-to-MAC address bindings in incoming ARP requests and ARP responses. Make sure to enable DHCP snooping to permit ARP packets that have dynamically assigned IP addresses.

Follow these steps to configure dynamic ARP inspection. You must perform this procedure on both switches. This procedure is required.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **show cdp neighbors**<br>**Example:**<br>`Device# show cdp neighbors` | Verify the connection between the switches. |
| **Step 3** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 4** | **ip arp inspection vlan** *vlan-range*<br>**Example:**<br>`Device(config)# ip arp inspection vlan 1` | Enable dynamic ARP inspection on a per-VLAN basis. By default, dynamic ARP inspection is disabled on all VLANs. For vlan-range, specify a single VLAN identified by VLAN ID number, a range of VLANs separated by a hyphen, or a series of VLANs |

| | Command or Action | Purpose |
|---|---|---|
| | | separated by a comma. The range is 1 to 4094. Specify the same VLAN ID for both switches. |
| **Step 5** | **Interface***type number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Specifies the interface connected to the other switch, and enter interface configuration mode. |
| **Step 6** | **ip arp inspection trust**<br>**Example:**<br>`Device(config-if)# ip arp inspection trust` | Configures the connection between the switches as trusted. By default, all interfaces are untrusted.<br>The switch does not check ARP packets that it receives from the other switch on the trusted interface. It simply forwards the packets.<br>For untrusted interfaces, the switch intercepts all ARP requests and responses. It verifies that the intercepted packets have valid IP-to-MAC address bindings before updating the local cache and before forwarding the packet to the appropriate destination. The switch drops invalid packets and logs them in the log buffer according to the logging configuration specified with the ip arp inspection vlan logging global configuration command. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 8** | **show ip arp inspection interfaces**<br>**Example:**<br>`Device# show ip arp inspection interfaces` | Verifies the dynamic ARP inspection configuration on interfaces. |
| **Step 9** | **show ip arp inspection vlan** *vlan-range*<br>**Example:**<br>`Device# show ip arp inspection vlan 1` | Verifies the dynamic ARP inspection configuration on VLAN. |
| **Step 10** | **show ip dhcp snooping binding**<br>**Example:**<br>`Device# show ip dhcp snooping binding` | Verifies the DHCP bindings. |
| **Step 11** | **show ip arp inspection statistics vlan** *vlan-range*<br>**Example:** | Checks the dynamic ARP inspection statistics on VLAN. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# show ip arp inspection statistics vlan 1` | |

# Limiting the Rate of Incoming ARP Packets

The switch CPU performs dynamic ARP inspection validation checks; therefore, the number of incoming ARP packets is rate-limited to prevent a denial- of-service attack.

When the rate of incoming ARP packets exceeds the configured limit, the switch places the port in the error-disabled state. The port remains in that state until you enable error-disabled recovery so that ports automatically emerge from this state after a specified timeout period.

**Note** Unless you configure a rate limit on an interface, changing the trust state of the interface also changes its rate limit to the default value for that trust state. After you configure the rate limit, the interface retains the rate limit even when its trust state is changed. If you enter the **no ip arp inspection limit** interface configuration command, the interface reverts to its default rate limit.

Follow these steps to limit the rate of incoming ARP packets. This procedure is optional.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 0/1/2` | Specifies the interface to be rate-limited, and enters interface configuration mode. |
| **Step 4** | **ip arp inspection limit {rate pps [burst interval seconds] \| none}** | Limits the rate of incoming ARP requests and responses on the interface. The default rate is 15 pps on untrusted interfaces and unlimited on trusted interfaces. The burst interval is 1 second.<br>The keywords have these meanings:<br>• For **rate***pps*, specify an upper limit for the number of incoming packets processed per second. The range is 0 to 2048 pps. |

| | Command or Action | Purpose |
|---|---|---|
| | | • (Optional) For **burst interval***seconds*, specify the consecutive interval in seconds, over which the interface is monitored for a high rate of ARP packets. The range is 1 to 15.<br>• For **rate none**, specify no upper limit for the rate of incoming ARP packets that can be processed. |
| **Step 5** | **exit**<br>**Example:**<br>`Device(config-if)# exit` | Exits interface configuration mode and returns to global configuration mode. |
| **Step 6** | Use the following commands:<br>• **errdisable detect cause arp-inspection**<br>• **errdisable recovery cause arp-inspection**<br>• **errdisable recovery interval** *interval*<br>**Example:**<br>`Device(config)# errdisable recovery cause arp-inspection` | (Optional) Enables error recovery from the dynamic ARP inspection error-disabled state, and configure the dynamic ARP inspection recover mechanism variables.<br>By default, recovery is disabled, and the recovery interval is 300 seconds.<br>For **interval** *interval*, specify the time in seconds to recover from the error-disabled state. The range is 30 to 86400. |
| **Step 7** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and returns to privileged EXEC mode. |

# Performing Dynamic ARP Inspection Validation Checks

Dynamic ARP inspection intercepts, logs, and discards ARP packets with invalid IP-to-MAC address bindings. You can configure the switch to perform additional checks on the destination MAC address, the sender and target IP addresses, and the source MAC address.

Follow these steps to perform specific checks on incoming ARP packets. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | **ip arp inspection validate {[src-mac] [dst-mac] [ip]}**<br>**Example:**<br>`Device(config)# ip inspection validate ip` | Performs a specific check on incoming ARP packets. By default, no checks are performed.<br>The keywords have these meanings:<br>• For **src-mac**, check the source MAC address in the Ethernet header against the sender MAC address in the ARP body. This check is performed on both ARP requests and responses. When enabled, packets with different MAC addresses are classified as invalid and are dropped.<br>• For **dst-mac**, check the destination MAC address in the Ethernet header against the target MAC address in ARP body. This check is performed for ARP responses. When enabled, packets with different MAC addresses are classified as invalid and are dropped.<br>• For **ip**, check the ARP body for invalid and unexpected IP addresses. Addresses include 0.0.0.0, 255.255.255.255, and all IP multicast addresses. Sender IP addresses are checked in all ARP requests and responses, and target IP addresses are checked only in ARP responses.<br>You must specify at least one of the keywords. Each command overrides the configuration of the previous command; that is, if a command enables src and dst mac validations, and a second command enables IP validation only, the src and dst mac validations are disabled as a result of the second command. |
| **Step 4** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and returns to privileged EXEC mode. |
| **Step 5** | **show ip arp inspection vlan** *vlan-range*<br>**Example:**<br>`Device# show ip arp insepction vlan 1-2` | Displays the statistics for the selected range of VLANs. |

# Monitoring DAI

To monitor DAI, use the following commands:

| Command | Description |
|---|---|
| **clear ip arp inspection statistics** | Clears dynamic ARP inspection statistics. |
| **show ip arp inspection statistics** [**vlan** *vlan-range*] | Displays statistics for forwarded, dropped, MAC validation failure, IP validation failure, ACL permitted and denied, and DHCP permitted and denied packets for the specified VLAN. If no VLANs are specified or if a range is specified, displays information only for VLANs with dynamic ARP inspection enabled (active). |
| **clear ip arp inspection log** | Clears the dynamic ARP inspection log buffer. |
| **show ip arp inspection log** | Displays the configuration and contents of the dynamic ARP inspection log buffer. |

For the **show ip arp inspection statistics** command, the switch increments the number of forwarded packets for each ARP request and response packet on a trusted dynamic ARP inspection port. The switch increments the number of ACL or DHCP permitted packets for each packet that is denied by source MAC, destination MAC, or IP validation checks, and the switch increments the appropriate.

# Verifying the DAI Configuration

To display and verify the DAI configuration, use the following commands:

| Command | Description |
|---|---|
| **show arp access-list** [*acl-name*] | Displays detailed information about ARP ACLs. |
| **show ip arp inspection interfaces** [interface-id] | Displays the trust state and the rate limit of ARP packets for the specified interface or all interfaces. |
| **show ip arp inspection vlan** *vlan-range* | Displays the configuration and the operating state of dynamic ARP inspection for the specified VLAN. If no VLANs are specified or if a range is specified, displays information only for VLANs with dynamic ARP inspection enabled (active). |

# Feature History for Dynamic ARP Inspection

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Dynamic ARP Inspection | ARP provides IP communication within a Layer 2 broadcast domain by mapping an IP address to a MAC address. Dynamic ARP inspection is a security feature that validates ARP packets in a network. It intercepts, logs,and discards ARP packets with invalid IP-to-MAC address bindings. This capability protects the network from certain man-in-the-middle attacks.<br><br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Dynamic ARP Inspection | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 26

# Configuring IPv6 First Hop Security


- Prerequisites for IPv6 First Hop Security, on page 437
- Restrictions for IPv6 First Hop Security, on page 437
- Information About IPv6 First Hop Security, on page 437
- How to Configure IPv6 First Hop Security, on page 441
- Configuration Examples for IPv6 First Hop Security, on page 467
- Additional References for IPv6 First Hop Security, on page 471
- Feature History for IPv6 First Hop Security, on page 471


## Prerequisites for IPv6 First Hop Security

You have configured the necessary IPv6 enabled SDM template.

## Restrictions for IPv6 First Hop Security

The following restrictions apply when applying FHS policies to EtherChannel interfaces (Port Channels):

- A physical port with an FHS policy attached cannot join an EtherChannel group.
- An FHS policy cannot be attached to an physical port when it is a member of an EtherChannel group.

## Information About IPv6 First Hop Security

IPv6 FHS is composed of the following IPv6 security features: IPv6 Snooping, IPv6 Neighbor Discovery Inspection, IPv6 Router Advertisement Guard , IPv6 DHCP Guard, IPv6 Source Guard, IPv6 Prefix Guard, IPv6 Destination Guard.

Each one of these security features addresses a different aspect of first hop security. In order to use a security feature, the corresponding policy must be configured. Policies specify a particular behavior. They must also be attached to a target, which can be a physical interface, an EtherChannel interface, or a VLAN. An IPv6 software policy database service stores and accesses these policies. When a policy is configured or modified, the attributes of the policy are stored or updated in the software policy database, and applied as specified.

In addition to the security features, the IPv6 FHS infrastructure has an IPv6 FHS Binding Table, which is a database table of IPv6 neighbors connected to the device. A binding entry includes information such as the

IP and MAC address of the host, the interface, VLAN, state of the entry etc. This database or binding table is used by other features (such as IPv6 ND Inspection) to validate the link-layer address (LLA), the IPv4 or IPv6 address, and prefix binding of the neighbors, to prevent spoofing and redirect attacks. The binding table is updated through the IPv6 Snooping feature, and through manually added static binding entries.

**Note** The IPv6 FHS Binding Table is supported through the Switch Integrated Security Feature (SISF) feature. For more information, see the *Configuring Switch Integrated Security Features* chapter in this guide.

### IPv6 Snooping

**Note** The IPv6 Snooping feature is deprecated and the SISF feature replaces it and offers the same capabilities. While the IPv6 Snooping commands are still available on the CLI and the existing configuration continues to be supported, the commands will be removed from the CLI in a later release. For more information about the replacement feature, see the *Configuring Switch Integrated Security Features* chapter in this guide.

IPv6 Snooping acts as a container that enables most of the features available with FHS in IPv6 including following capabilities and functions:

- Neighbor Discovery Snooping: IPv6 Neighbor Discovery Snooping analyzes and verifies IPv6 Neighbor Discovery Protocol (NDP) traffic. During inspection, it gleans address bindings (IP, MAC, port, etc) and stores it in the binding table.
- DHCPv6 Snooping: DHCPv6 Snooping traps DHCPv6 packets between DHCPv6 Client and DHCPv6 Server. From the packets snooped, assigned addresses are learnt and stored in the binding table.
- Device tracking: IPv6 Snooping also tracks the movement of hosts from one port to another, verifies their existence using Duplicate Address Detection (DAD).
- With the IPv6 Snooping feature one can limit the number of addresses any node on the link can claim. This feature can be used to protect the switch binding table against denial of service flooding attacks.

By default, a snooping policy has a security-level of guard. When a snooping policy is configured on an access switch, external IPv6 Router Advertisement (RA) or Dynamic Host Configuration Protocol for IPv6 (DHCPv6) server packets are blocked, even though the uplink port facing the device or DHCP server or relay is configured as a trusted port. To allow IPv6 RA or DHCPv6 server messages, do the following:

- Apply an IPv6 RA-guard policy (for RA) or IPv6 DHCP-guard policy (for DHCP server messages) on the uplink port.
- Configure a snooping policy with a lower security-level, for example glean or inspect. This is a less preferable option, because the benefits of FHS features are not effective.

To use this feature, configure an IPv6 Snooping policy and attach it to a target. See Configuring an IPv6 Snooping Policy, on page 441.

### IPv6 Neighbor Discovery Inspection

**Note** Starting with Cisco IOS XE Amsterdam 17.1.1, the IPv6 Neighbor Discovery Inspection (IPv6 ND Inspection) feature is deprecated and the SISF feature replaces it and offers the same capabilities. While the IPv6 ND Inspection commands are still available on the CLI and the existing configuration continues to be supported, the commands will be removed from the CLI in a later release. For more information about the replacement feature, see the *Configuring Switch Integrated Security Features* chapter in this guide.

The IPv6 ND Inspection feature learns and secures bindings for stateless auto-configuration addresses in Layer 2 neighbor tables. It analyzes neighbor discovery messages in order to build a trusted binding table database. IPv6 neighbor discovery messages that do not conform are dropped. An neighbor discovery message is considered trustworthy if its IPv6-to-MAC mapping is verifiable.

This feature mitigates some of the inherent vulnerabilities of the ND mechanism, such as attacks on DAD, address resolution, router discovery, and the neighbor cache.

To use this feature, configure an IPv6 ND Inspection policy and attach it to a target. See Configuring an IPv6 Neighbor Discovery Inspection Policy, on page 447.

### IPv6 Router Advertisement Guard

This feature enables the network administrator to block or reject unwanted or rogue Router Advertisement (RA) guard messages that arrive at the network device platform. RAs are used by devices to announce themselves on the link. The RA Guard feature analyzes the RAs and filters out bogus RAs sent by unauthorized devices. In host mode, all router advertisement and router redirect messages are disallowed on the port. The RA guard feature compares configuration information on the Layer 2 device with the information found in the received RA frame. Once the Layer 2 device has validated the content of the RA frame and router redirect frame against the configuration, it forwards the RA to its unicast or multicast destination. If the RA frame content is not validated, the RA is dropped.

SISF-based device-tracking forwards router solicitation packets only on interfaces that have the RA guard policy configured and are also designated as router-facing interfaces. If no such interface exists, the router solicitation messages are dropped, which might delay the router discovery for onboarding hosts as they will be unable to discover the router until it sends a periodic unsolicited router advertisement.

To use this feature, configure an IPv6 RA Guard policy and attach it to a target. See Configuring an IPv6 Router Advertisement Guard Policy, on page 452.

### IPv6 DHCP Guard

The IPv6 DHCP Guard feature blocks reply and advertisement messages that come from unauthorized DHCPv6 servers and relay agents. IPv6 DHCP guard can prevent forged messages from being entered in the binding table and block DHCPv6 server messages when they are received on ports that are not explicitly configured as facing a DHCPv6 server or DHCP relay.

To use this feature, configure an IPv6 DHCP Guard policy and attach it to a target. See Configuring an IPv6 DHCP Guard Policy , on page 457.

To debug DHCP guard packets, use the **debug ipv6 snooping dhcp-guard** privileged EXEC command.

### IPv6 Source Guard

The IPv6 Source Guard feature validates the source of IPv6 traffic to prevent source address spoofing. It deals exclusively with data packet traffic. You can use this feature to deny traffic from unknown sources, traffic from sources not assigned by a DHCP server, etc.

It involves a hardware-programmed (TCAM table) filter which allows or denies traffic based on its source address. For the filter to work this way, an entry (of the source address) in the binding table is required. If the source address is in the binding table, the filter allows the packet into the network; if the address is not in the binding table, entry is denied and the packet is dropped. When an entry is removed from the binding table, the filter is also removed, and subsequent packets with that source address are dropped.

When configuring this feature, consider the following:

- The IPv6 Source Guard and Prefix Guard features are supported only in the ingress direction and not supported in the egress direction.
- You cannot use IPv6 Source Guard and Prefix Guard together. When you attach the policy to an interface, it should be "validate address" or "validate prefix" but not both.
- PVLAN and Source or Prefix Guard cannot be applied together.
- IPv6 Source Guard and Prefix Guard is supported on EtherChannels
- An IPv6 source guard policy cannot be attached to a VLAN. It is supported only at the interface level.
- When you configure IPv4 and IPv6 source guard together on an interface, it is recommended to use **ip verify source mac-check** command instead of **ip verify source tracking mac-check** command. IPv4 connectivity on a given port might break due to two different filtering rules set: one for IPv4 (IP-filter) and the other for IPv6 (IP-MAC filter).
- When IPv6 source guard is enabled on a switch port, NDP or DHCP snooping must be enabled on the interface to which the switch port belongs. Otherwise, all data traffic from this port will be blocked.
- Binding information is normally gleaned from IPv6 NDP traffic and DHCP packets. If you rely only on a DHCP server for source addresses of hosts, ensure that you also configure a data-glean recovery function to counteract a situation where entries are prematurely removed from the binding table (for various reasons) before the DHCP lease timer expires. This way, the recovery function *restores* binding entries of valid hosts and you can be sure that that the IPv6 Source Guard feature allows only packets with a DHCP server-assigned source address. See Example: Using the Data-Glean Recovery Function, on page 468.

To use this feature, you must configure an IPv6 Source Guard policy and attach it to a target. See Configuring IPv6 Source Guard, on page 461 .

To debug source-guard packets, use the **debug ipv6 snooping source-guard** privileged EXEC command.

### IPv6 Prefix Guard

The IPv6 Prefix Guard feature works within the IPv6 Source Guard feature to enable the device to deny traffic originated from non-topologically correct addresses. IPv6 Prefix Guard is often used when IPv6 prefixes are delegated to devices (for example, home gateways) using DHCP prefix delegation. The feature discovers ranges of addresses assigned to the link and blocks any traffic sourced with an address outside this range.

In order to use this feature, you must configure an IPv6 Prefix Guard policy and attach it to a target. See Configuring IPv6 Prefix Guard, on page 464.

**Note** Ensure that you have read the configuration considerations listed in the **IPv6 Source Guard** section above - some of them apply to the IPv6 Prefix Guard feature as well.

#### IPv6 Destination Guard

The IPv6 Destination Guard feature works with IPv6 neighbor discovery to ensure that the device performs address resolution only for those addresses that are known to be active on the link. It relies on the address glean functionality to populate all destinations active on the link into the binding table and then blocks resolutions before they happen when the destination is not found in the binding table.

**Note** We recommend that you apply an IPv6 Destination Guard policy on all Layer 2 VLANs with an SVI configured.

In order to use this feature, you must configure an IPv6 Destination Guard policy and attach it to a target. See Configuring an IPv6 Destination Guard Policy, on page 466.

# How to Configure IPv6 First Hop Security

## Configuring an IPv6 Snooping Policy

**Note** The IPv6 Snooping Policy feature has been deprecated. Although the commands are visible on the CLI and you can configure them, we recommend that you use the Switch Integrated Security Feature (SISF)-based Device Tracking feature instead.

Beginning in privileged EXEC mode, follow these steps to configure IPv6 Snooping Policy :

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ipv6 snooping policy** *policy-name*<br>**Example:** | Creates a snooping policy and enters IPv6 snooping policy configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# ipv6 snooping policy example_policy` | |
| **Step 4** | {[**default** ] \| [**device-role** {**node** \| **switch**}] \| [**limit address-count** *value*] \| [**no**] \| [**protocol** {**dhcp** \| **ndp**} ] \| [**security-level** {**glean** \| **guard** \| **inspect**} ] \| [**tracking** {**disable** [**stale-lifetime** [*seconds* \| **infinite**] \| **enable** [**reachable-lifetime** [*seconds* \| **infinite**] } ] \| [**trusted-port** ] }<br>**Example:**<br>`Device(config-ipv6-snooping)# security-level inspect`<br>**Example:**<br>`Device(config-ipv6-snooping)# trusted-port` | Enables data address gleaning, validates messages against various criteria, specifies the security level for messages.<br>• (Optional) **default**: Sets all to default options.<br>• (Optional) **device-role**{**node**] \| **switch**}: Specifies the role of the device attached to the port. Default is **node**.<br>• (Optional) **limit address-count** *value*: Limits the number of addresses allowed per target.<br>• (Optional) **no**: Negates a command or sets it to defaults.<br>• (Optional) **protocol**{**dhcp** \| **ndp**}: Specifies which protocol should be redirected to the snooping feature for analysis. The default, is **dhcp** and **ndp**. To change the default, use the **no protocol** command.<br>• (Optional) **security-level**{**glean**\|**guard**\|**inspect**}: Specifies the level of security enforced by the feature. Default is **guard.**<br>**glean**: Gleans addresses from messages and populates the binding table without any verification.<br>**guard**: Gleans addresses and inspects messages. In addition, it rejects RA and DHCP server messages. This is the default option.<br>**inspect**: Gleans addresses, validates messages for consistency and conformance, and enforces address ownership.<br>• (Optional) **tracking** {**disable** \| **enable**}: Overrides the default tracking behavior and specifies a tracking option.<br>• (Optional) **trusted-port**: Sets up a trusted port. It disables the guard on applicable targets. Bindings learned through a trusted port have preference over bindings learned through any other port. A trusted port is |

| | Command or Action | Purpose |
|---|---|---|
| | | given preference in case of a collision while making an entry in the table. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-ipv6-snooping)# end` | Exits IPv6 snooping policy configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show ipv6 snooping policy** *policy-name*<br>**Example:**<br>`Device#show ipv6 snooping policy example_policy` | Displays the snooping policy configuration. |

### What to do next

Attach an IPv6 Snooping policy to interfaces or VLANs.

# Attaching an IPv6 Snooping Policy to an Interface

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 Snooping policy on an interface or VLAN:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface_type stack/module/port*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/1/4` | Specifies an interface type and identifier and enters the interface configuration mode. |
| **Step 4** | **switchport**<br>**Example:**<br>`Device(config-if)# switchport` | Enters the Switchport mode.<br>**Note**<br>To configure Layer 2 parameters, if the interface is in Layer 3 mode, you must enter the **switchport** interface configuration command without any parameters to change the interface into Layer 2 mode. This shuts down the interface and then re-enables it, |

| | Command or Action | Purpose |
|---|---|---|
| | | which might generate messages on the device to which the interface is connected. When change the interface mode from Layer 3 to Layer 2 mode, the previous configuration information related to the affected interface might be lost, and the interface is returned to its default configuration. The command prompt displays as (config-if)# in Switchport configuration mode. |
| **Step 5** | **ipv6 snooping** [**attach-policy** *policy_name* [ **vlan** {*vlan_id* \| **add** *vlan_ids* \| **except** *vlan_ids* \| **none** \| **remove** *vlan_ids*}] \| **vlan** {*vlan_id* \| **add** *vlan_ids* \| **except** *vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ]<br>**Example:**<br>`Device(config-if)# ipv6 snooping`<br>`Device(config-if)# ipv6 snooping attach-policy example_policy`<br>`Device(config-if)# ipv6 snooping vlan 111,112`<br>`Device(config-if)# ipv6 snooping attach-policy example_policy vlan 111,112` | Attaches a custom IPv6 snooping policy to the interface or the specified VLANs on the interface. To attach the default policy to the interface, use the **ipv6 snooping** command without the **attach-policy** keyword. To attach the default policy to VLANs on the interface, use the **ipv6 snooping vlan** command. The default policy is, security-level **guard**, device-role **node**, protocol **ndp** and **dhcp.** |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 7** | **show running-config**<br>**Example:**<br>`Device# show running-config` | Verifies that the policy is attached to the specified interface without exiting the interface configuration mode. |

# Attaching an IPv6 Snooping Policy to a Layer 2 EtherChannel Interface

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 Snooping policy on an EtherChannel interface or VLAN:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface range** *interface_name*<br>**Example:**<br>`Device(config)# interface range Port-channel 11` | Specifies the port-channel interface name assigned when the EtherChannel was created. Enters the interface range configuration mode.<br>**Tip**<br>Enter the **show interfaces summary** command for quick reference to interface names and types. |
| **Step 4** | **ipv6 snooping** [**attach-policy** *policy_name* [ **vlan** {*vlan_ids* \| **add** *vlan_ids* \| **except** *vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ] \| **vlan** [ {*vlan_ids* \| **add** *vlan_ids* \| **except***vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ]<br>**Example:**<br>`Device(config-if-range)# ipv6 snooping attach-policy example_policy`<br>`Device(config-if-range)# ipv6 snooping attach-policy example_policy vlan 222,223,224`<br>`Device(config-if-range)# ipv6 snooping vlan 222, 223,224` | Attaches the IPv6 Snooping policy to the interface or the specified VLANs on that interface. The default policy is attached if the **attach-policy** option is not used. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if-range)# end` | Exits interface range configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show running-config interface***portchannel_interface_name*<br>**Example:**<br>`Device# show running-config interface portchannel 11` | Confirms that the policy is attached to the specified interface. |

# Attaching an IPv6 Snooping Policy to VLANs Globally

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 Snooping Policy to VLANs across multiple interfaces:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** | Enables privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device> enable` | Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **vlan configuration** *vlan_list*<br>**Example:**<br>`Device(config)# vlan configuration 333` | Specifies the VLANs to which the IPv6 Snooping policy will be attached, and enters the VLAN interface configuration mode. |
| **Step 4** | **ipv6 snooping** [**attach-policy** *policy_name*]<br>**Example:**<br>`Device(config-vlan-config)#ipv6 snooping attach-policy example_policy` | Attaches the IPv6 Snooping policy to the specified VLANs across all device interfaces. The default policy is attached if the **attach-policy** option is not used. The default policy is, security-level **guard**, device-role **node**, protocol **ndp** and **dhcp.** |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-vlan-config)# end` | Exits VLAN interface configuration mode and returns to privileged EXEC mode. |

# Configuring the IPv6 Binding Table Content

Beginning in privileged EXEC mode, follow these steps to configure IPv6 Binding Table Content :

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | [**no**] **ipv6 neighbor binding** [**vlan** *vlan-id* {*ipv6-address* **interface** interface_type *stack/module/port hw_address* [**reachable-lifetimevalue** [*seconds* \| **default** \| **infinite**] \| [**tracking**{ [default \| disable] [ **reachable-lifetimevalue** [*seconds* \| **default** \| **infinite**] \| [**enable** [**reachable-lifetimevalue** | Adds a static entry to the binding table database. |

| | Command or Action | Purpose |
|---|---|---|
| | [*seconds* \| **default** \| **infinite**] \| [**retry-interval** {*seconds*\| **default** [**reachable-lifetimevalue** [*seconds* \| **default** \| **infinite**] } ]<br>**Example:**<br>`Device(config)# ipv6 neighbor binding` | |
| **Step 4** | [**no**] **ipv6 neighbor binding max-entries** *number* [**mac-limit** *number* \| **port-limit** *number* [**mac-limit** *number*] \| **vlan-limit** *number* [ [**mac-limit** *number*] \| [**port-limit** *number* [**mac-limit***number*] ] ] ]<br>**Example:**<br>`Device(config)# ipv6 neighbor binding max-entries 30000` | Specifies the maximum number of entries that are allowed to be inserted in the binding table cache. |
| **Step 5** | **ipv6 neighbor binding logging**<br>**Example:**<br>`Device(config)# ipv6 neighbor binding logging` | Enables the logging of binding table main events. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and returns to privileged EXEC mode. |
| **Step 7** | **show ipv6 neighbor binding**<br>**Example:**<br>`Device# show ipv6 neighbor binding` | Displays contents of a binding table. |

# Configuring an IPv6 Neighbor Discovery Inspection Policy

Starting with Cisco IOS XE Amsterdam 17.1.1 the IPv6 ND Inspection feature is deprecated and the SISF-based device tracking feature replaces it and offers the same capabilities. For the corresponding replacement task, see *Creating a Custom Device Tracking Policy with Custom Settings* under the *Configuring SISF-Based Device Tracking* chapter in this document.

Beginning in privileged EXEC mode, follow these steps to configure an IPv6 ND Inspection Policy:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ipv6 nd inspection policy** *policy-name*<br>**Example:**<br>`Device(config)# ipv6 nd inspection policy example_policy` | Specifies the ND inspection policy name and enters ND Inspection Policy configuration mode. |
| **Step 4** | **device-role** {**host** \| **switch**}<br>**Example:**<br>`Device(config-nd-inspection)# device-role switch` | Specifies the role of the device attached to the port. The default is **host**. |
| **Step 5** | **limit address-count** *value*<br>**Example:**<br>`Device(config-nd-inspection)# limit address-count 1000` | Limits the number of IPv6 addresses allowed to be used on the port. |
| **Step 6** | **tracking** {**enable** [**reachable-lifetime** {*value* \| **infinite**}] \| **disable** [**stale-lifetime** {*value* \| **infinite**}]}<br>**Example:**<br>`Device(config-nd-inspection)# tracking disable stale-lifetime infinite` | Overrides the default tracking policy on a port. |
| **Step 7** | **trusted-port**<br>**Example:**<br>`Device(config-nd-inspection)# trusted-port` | Configures a port to become a trusted port. |
| **Step 8** | **validate source-mac**<br>**Example:**<br>`Device(config-nd-inspection)# validate source-mac` | Checks the source media access control (MAC) address against the link-layer address. |
| **Step 9** | **no** {**device-role** \| **limit address-count** \| **tracking** \| **trusted-port** \| **validate source-mac**}<br>**Example:**<br>`Device(config-nd-inspection)# no validate source-mac` | Removes the current configuration of a parameter with the **no** form of the command. |
| **Step 10** | **default** {**device-role** \| **limit address-count** \| **tracking** \| **trusted-port** \| **validate source-mac**}<br>**Example:** | Restores configuration to the default values. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-nd-inspection)# default limit address-count` | |
| **Step 11** | **end**<br>**Example:**<br>`Device(config-nd-inspection)# end` | Exits ND Inspection Policy configuration mode and returns to privileged EXEC mode. |
| **Step 12** | **show ipv6 nd inspection policy** *policy_name*<br>**Example:**<br>`Device# show ipv6 nd inspection policy example_policy` | Verifies the ND inspection configuration. |

## Attaching an IPv6 Neighbor Discovery Inspection Policy to an Interface

Starting with Cisco IOS XE Amsterdam 17.1.1 the IPv6 ND Inspection feature is deprecated and the SISF-based device tracking feature replaces it and offers the same capabilities. For the corresponding replacement task, see *Attaching a Device Tracking Policy to an Interface* under the *Configuring SISF-Based Device Tracking* chapter in this document.

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 ND Inspection policy to an interface or VLANs on an interface :

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-type interface-number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/1/4` | Specifies an interface type and identifier; enters the interface configuration mode. |
| **Step 4** | **ipv6 nd inspection** [**attach-policy** *policy_name* [ **vlan** {*vlan_ids* \| **add** *vlan_ids* \| **except** *vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ] \| **vlan** [ {*vlan_ids* \| **add** *vlan_ids* \| **except***vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ]<br>**Example:**<br>`Device(config-if)# ipv6 nd inspection attach-policy example_policy` | Attaches the Neighbor Discovery Inspection policy to the interface or the specified VLANs on that interface. The default policy is attached if the **attach-policy** option is not used. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-if)# ipv6 nd inspection attach-policy example_policy vlan 222,223,2`<br>`Device(config-if)# ipv6 nd inspection vlan 222, 223,224` | |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

## Attaching an IPv6 Neighbor Discovery Inspection Policy to a Layer 2 EtherChannel Interface

Starting with Cisco IOS XE Amsterdam 17.1.1 the IPv6 ND Inspection feature is deprecated and the SISF-based device tracking feature replaces it and offers the same capabilities. For the corresponding replacement task, see *Attaching a Device Tracking Policy to an Interface* under the *Configuring SISF-Based Device Tracking* chapter in this document.

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 Neighbor Discovery Inspection policy on an EtherChannel interface or VLAN:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface range** *interface_name*<br>**Example:**<br>`Device(config)# interface range Port-channel 11` | Specifies the port-channel interface name assigned when the EtherChannel was created. Enters interface range configuration mode.<br>**Tip**<br>Enter the **show interfaces summary** command for quick reference to interface names and types. |
| **Step 4** | **ipv6 nd inspection** [**attach-policy** *policy_name* [ **vlan** {*vlan_ids* \| **add** *vlan_ids* \| **except** *vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ] \| **vlan** [ {*vlan_ids* \| **add** *vlan_ids* \| **except***vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ]<br>**Example:** | Attaches the ND Inspection policy to the interface or the specified VLANs on that interface. The default policy is attached if the **attach-policy** option is not used. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-if-range)# ipv6 nd inspection attach-policy example_policy`<br>`Device(config-if-range)# ipv6 nd inspection vlan 222, 223,224`<br>`Device(config-if-range)# ipv6 nd inspection attach-policy example_policy vlan 222,223,224` | |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if-range)# end` | Exits interface range configuration mode and returns to privileged EXEC mode. |

## Attaching an IPv6 Neighbor Discovery Inspection Policy to VLANs Globally

Starting with Cisco IOS XE Amsterdam 17.1.1 the IPv6 ND Inspection feature is deprecated and the SISF-based device tracking feature replaces it and offers the same capabilities. For the corresponding replacement task, see *Attaching a Device Tracking Policy to a VLAN* under the *Configuring SISF-Based Device Tracking* chapter in this document.

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 ND Inspection policy to VLANs across multiple interfaces:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **vlan configuration** *vlan_list*<br>**Example:**<br>`Device(config)# vlan configuration 334` | Specifies the VLANs to which the IPv6 Snooping policy will be attached, and enters VLAN interface configuration mode. |
| **Step 4** | **ipv6 nd inspection** [**attach-policy** *policy_name*]<br>**Example:**<br>`Device(config-vlan-config)#ipv6 nd inspection attach-policy example_policy` | Attaches the IPv6 Neighbor Discovery policy to the specified VLANs across all switch and stack interfaces. The default policy is attached if the **attach-policy** option is not used.<br>The default policy is, device-role **host**, no drop-unsecure, limit address-count disabled, sec-level minimum is disabled, tracking is disabled, no trusted-port, no validate source-mac. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br>**Example:**<br>`Device(config-vlan-config)# end` | Exits VLAN interface configuration mode and returns to privileged EXEC mode. |

# Configuring an IPv6 Router Advertisement Guard Policy

Beginning in privileged EXEC mode, follow these steps to configure an IPv6 Router Advertisement policy :

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ipv6 nd raguard policy** *policy-name*<br>**Example:**<br>`Device(config)# ipv6 nd raguard policy example_policy` | Specifies the RA guard policy name and enters RA guard policy configuration mode. |
| **Step 4** | [**no**]**device-role** {**host** \| **monitor** \| **router** \| **switch**}<br>**Example:**<br>`Device(config-nd-raguard)# device-role switch` | Specifies the role of the device attached to the port. The default is **host**.<br>**Note**<br>For a network with both host-facing ports and router-facing ports, along with a RA guard policy configured with **device-role host** on host-facing ports or vlan, it is mandatory to configure a RA guard policy with **device-role router** on router-facing ports to allow the RA Guard feature to work properly. |
| **Step 5** | **hop-limit** {**maximum** \| **minimum**} *value*<br>**Example:**<br>`Device(config-nd-raguard)# hop-limit maximum 33` | Enables filtering of Router Advertisement messages by the Hop Limit value. A rogue RA message may have a low Hop Limit value (equivalent to the IPv4 Time to Live) that when accepted by the host, prevents the host from generating traffic to destinations beyond the rogue RA message generator. An RA message with an unspecified Hop Limit value is blocked. |

| | Command or Action | Purpose |
|---|---|---|
| | | (1–255) Range for Maximum and Minimum Hop Limit values.<br>If not configured, this filter is disabled. Configure **minimum** to block RA messages with Hop Limit values lower than the value you specify. Configure **maximum**to block RA messages with Hop Limit values greater than the value you specify. |
| **Step 6** | **managed-config-flag** {**off** \| **on**}<br>**Example:**<br>`Device(config-nd-raguard)# managed-config-flag on` | Enables filtering of Router Advertisement messages by the managed address configuration, or "M" flag field. A rouge RA message with an M field of 1 can cause a host to use a rogue DHCPv6 server. If not configured, this filter is disabled.<br>**On**: Accepts and forwards RA messages with an M value of 1, blocks those with 0.<br>**Off**: Accepts and forwards RA messages with an M value of 0, blocks those with 1. |
| **Step 7** | **match** {**ipv6 access-list** *list* \| **ra prefix-list** *list*}<br>**Example:**<br>`Device(config-nd-raguard)# match ipv6 access-list example_list` | Matches a specified prefix list or access list. |
| **Step 8** | **other-config-flag** {**on** \| **off**}<br>**Example:**<br>`Device(config-nd-raguard)# other-config-flag on` | Enables filtering of Router Advertisement messages by the Other Configuration, or "O" flag field. A rouge RA message with an O field of 1 can cause a host to use a rogue DHCPv6 server. If not configured, this filter is disabled.<br>**On**: Accepts and forwards RA messages with an O value of 1, blocks those with 0.<br>**Off**: Accepts and forwards RA messages with an O value of 0, blocks those with 1. |
| **Step 9** | [**no**]**router-preference maximum** {**high** \| **medium** \| **low**}<br>**Example:**<br>`Device(config-nd-raguard)# router-preference maximum high` | Enables filtering of Router Advertisement messages by the router preference flag. If not configured, this filter is disabled.<br>• **high**: Accepts RA messages with the router preference set to high, medium, or low.<br>• **medium**: Blocks RA messages with the router preference set to high.<br>• **low**: Blocks RA messages with the router preference set to medium and high. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 10** | **trusted-port**<br>**Example:**<br>`Device(config-nd-raguard)# trusted-port` | When configured as a trusted port, all attached devices are trusted, and no further message verification is performed. |
| **Step 11** | **default** {**device-role** \| **hop-limit** {**maximum** \| **minimum**} \| **managed-config-flag** \| **match** {**ipv6 access-list** \| **ra prefix-list** } \| **other-config-flag** \| **router-preference maximum**\| **trusted-port**}<br>**Example:**<br>`Device(config-nd-raguard)# default hop-limit` | Restores a command to its default value. |
| **Step 12** | **end**<br>**Example:**<br>`Device(config-nd-raguard)# end` | Exits RA Guard policy configuration mode and returns to privileged EXEC mode. |
| **Step 13** | **show ipv6 nd raguard policy** *policy_name*<br>**Example:**<br>`Device# show ipv6 nd raguard policy example_policy` | (Optional) Displays the ND guard policy configuration. |

## Attaching an IPv6 Router Advertisement Guard Policy to an Interface

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 Router Advertisement policy to an interface or to VLANs on the interface :

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/1/4` | Specifies an interface type and identifier; enters the interface configuration mode. |
| **Step 4** | **ipv6 nd raguard** [**attach-policy** *policy_name* [ **vlan** {*vlan_ids* \| **add** *vlan_ids* \| **except** | Attaches the Neighbor Discovery Inspection policy to the interface or the specified VLANs |

| | Command or Action | Purpose |
|---|---|---|
| | *vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ] \| **vlan** [ {*vlan_ids* \| **add** *vlan_ids* \| **except***vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ]<br>**Example:**<br>`Device(config-if)# ipv6 nd raguard attach-policy example_policy`<br>`Device(config-if)# ipv6 nd raguard attach-policy example_policy vlan 222,223,224`<br>`Device(config-if)# ipv6 nd raguard vlan 222, 223,224` | on that interface. The default policy is attached if the **attach-policy** option is not used. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

## Attaching an IPv6 Router Advertisement Guard Policy to a Layer 2 EtherChannel Interface

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 Router Advertisement Guard Policy on an EtherChannel interface or VLAN:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface range** *type number*<br>**Example:**<br>`Device(config)# interface Port-channel 11` | Specifies the port-channel interface name assigned when the EtherChannel was created. Enters interface range configuration mode.<br>**Tip**<br>Enter the **show interfaces summary** command in privileged EXEC mode for quick reference to interface names and types. |
| **Step 4** | **ipv6 nd raguard** [**attach-policy** *policy_name* [ **vlan** {*vlan_ids* \| **add** *vlan_ids* \| **except** *vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ] \| **vlan** [ {*vlan_ids* \| **add** *vlan_ids* \| **except***vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ] | Attaches the RA Guard policy to the interface or the specified VLANs on that interface. The default policy is attached if the **attach-policy** option is not used. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device(config-if-range)# ipv6 nd raguard attach-policy example_policy`<br>`Device(config-if-range)# ipv6 nd raguard attach-policy example_policy vlan 222,223,224`<br>`Device(config-if-range)# ipv6 nd raguard vlan 222, 223,224` | |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if-range)# end` | Exits interface range configuration mode and returns to privileged EXEC mode. |

## Attaching an IPv6 Router Advertisement Guard Policy to VLANs Globally

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 Router Advertisement policy to VLANs regardless of interface:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **vlan configuration** *vlan_list*<br>**Example:**<br>`Device(config)# vlan configuration 335` | Specifies the VLANs to which the IPv6 RA Guard policy will be attached, and enters VLAN interface configuration mode. |
| **Step 4** | **ipv6 dhcp guard** [**attach-policy** *policy_name*]<br>**Example:**<br>`Device(config-vlan-config)# ipv6 nd raguard attach-policy example_policy` | Attaches the IPv6 RA Guard policy to the specified VLANs across all switch and stack interfaces. The default policy is attached if the **attach-policy** option is not used. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-vlan-config)# end` | Exits VLAN interface configuration mode and returns to privileged EXEC mode. |

# Configuring an IPv6 DHCP Guard Policy

Beginning in privileged EXEC mode, follow these steps to configure an IPv6 DHCP (DHCPv6) Guard policy:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ipv6 dhcp guard policy** *policy-name*<br>**Example:**<br>`Device(config)# ipv6 dhcp guard policy example_policy` | Specifies the DHCPv6 Guard policy name and enters DHCPv6 Guard Policy configuration mode. |
| **Step 4** | **device-role** {**client** \| **server**}<br>**Example:**<br>`Device(config-dhcp-guard)# device-role server` | (Optional) Filters out DHCPv6 replies and DHCPv6 advertisements on the port that are not from a device of the specified role. Default is **client**.<br>• **client**: Default value, specifies that the attached device is a client. Server messages are dropped on this port.<br>• **server**: Specifies that the attached device is a DHCPv6 server. Server messages are allowed on this port. |
| **Step 5** | **match server access-list** *ipv6-access-list-name*<br>**Example:**<br>`;;Assume a preconfigured IPv6 Access List as follows:`<br>`Device(config)# ipv6 access-list my_acls`<br>`Device(config-ipv6-acl)# permit host 2001:BD8:::1 any`<br><br>`;;configure DCHPv6 Guard to match approved access list.`<br>`Device(config-dhcp-guard)# match server access-list my_acls` | (Optional). Enables verification that the advertised DHCPv6 server or relay address is from an authorized server access list (The destination address in the access list is 'any'). If not configured, this check will be bypassed. An empty access list is treated as a permit all. |
| **Step 6** | **match reply prefix-list** *ipv6-prefix-list-name*<br>**Example:**<br><br>`;;Assume a preconfigured IPv6 prefix` | (Optional) Enables verification of the advertised prefixes in DHCPv6 reply messages from the configured authorized prefix list. If not configured, this check will be bypassed. An empty prefix list is treated as a permit. |

| | Command or Action | Purpose |
|---|---|---|
| | list as follows:<br>Device(config)# **ipv6 prefix-list my_prefix permit 2001:DB8::/64 le 128**<br><br>;; Configure DCHPv6 Guard to match prefix<br>Device(config-dhcp-guard)# **match reply prefix-list my_prefix** | |
| **Step 7** | **preference** { **max** *limit* \| **min** *limit* }<br>**Example:**<br>Device(config-dhcp-guard)# **preference max 250**<br>Device(config-dhcp-guard)#**preference min 150** | Configure **max** and **min** when **device-role** is **server**to filter DCHPv6 server advertisements by the server preference value. The defaults permit all advertisements.<br>**max** *limit*—(0 to 255) (Optional) Enables verification that the advertised preference (in preference option) is less than the specified limit. Default is 255. If not specified, this check will be bypassed.<br>**min** *limit*—(0 to 255) (Optional) Enables verification that the advertised preference (in preference option) is greater than the specified limit. Default is 0. If not specified, this check will be bypassed. |
| **Step 8** | **trusted-port**<br>**Example:**<br>Device(config-dhcp-guard)# **trusted-port** | (Optional) **trusted-port**—Sets the port to a trusted mode. No further policing takes place on the port.<br>**Note**<br>If you configure a trusted port then the device-role option is not available. |
| **Step 9** | **default** {**device-role** \| **trusted-port**}<br>**Example:**<br>Device(config-dhcp-guard)# **default device-role** | (Optional) **default**—Sets a command to its defaults. |
| **Step 10** | **end**<br>**Example:**<br>Device(config-dhcp-guard)# **end** | Exits DHCPv6 Guard Policy configuration mode and returns to privileged EXEC mode. |
| **Step 11** | **show ipv6 dhcp guard policy** *policy_name*<br>**Example:**<br>Device# **show ipv6 dhcp guard policy example_policy** | (Optional) Displays the configuration of the IPv6 DHCP guard policy. Omitting the *policy_name* variable displays all DHCPv6 policies. |

## Attaching an IPv6 DHCP Guard Policy to an Interface or a VLAN on an Interface

Beginning in privileged EXEC mode, follow these steps to configure IPv6 Binding Table Content :

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/1/4` | Specifies an interface type and identifier, and enters interface configuration mode. |
| **Step 4** | **ipv6 dhcp guard** [**attach-policy** *policy_name* [ **vlan** {*vlan_ids* \| **add** *vlan_ids* \| **except** *vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ] \| **vlan** [ {*vlan_ids* \| **add** *vlan_ids* \| **except***vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ]<br>**Example:**<br>`Device(config-if)# ipv6 dhcp guard attach-policy example_policy`<br>`Device(config-if)# ipv6 dhcp guard attach-policy example_policy vlan 222,223,224`<br>`Device(config-if)# ipv6 dhcp guard vlan 222, 223,224` | Attaches the DHCP Guard policy to the interface or the specified VLANs on that interface. The default policy is attached if the **attach-policy** option is not used. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

## Attaching an IPv6 DHCP Guard Policy to a Layer 2 EtherChannel Interface

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 DHCP Guard policy on an EtherChannel interface or VLAN:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface range** *Interface_name*<br>**Example:**<br>`Device(config)# interface Port-channel 11` | Specify the port-channel interface name assigned when the EtherChannel was created. Enters interface range configuration mode.<br>**Tip**<br>Enter the **show interfaces summary** command in privileged EXEC mode for quick reference to interface names and types. |
| **Step 4** | **ipv6 dhcp guard** [**attach-policy** *policy_name* [ **vlan** {*vlan_ids* \| **add** *vlan_ids* \| **except** *vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ] \| **vlan** [ {*vlan_ids* \| **add** *vlan_ids* \| **except***vlan_ids* \| **none** \| **remove** *vlan_ids* \| **all**} ]<br>**Example:**<br>`Device(config-if-range)# ipv6 dhcp guard attach-policy example_policy`<br>`Device(config-if-range)# ipv6 dhcp guard attach-policy example_policy vlan 222,223,224`<br>`Device(config-if-range)# ipv6 dhcp guard vlan 222, 223,224` | Attaches the DHCP Guard policy to the interface or the specified VLANs on that interface. The default policy is attached if the **attach-policy** option is not used. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if-range)# end` | Exits interface range configuration mode and returns to privileged EXEC mode. |

## Attaching an IPv6 DHCP Guard Policy to VLANs Globally

Beginning in privileged EXEC mode, follow these steps to attach an IPv6 DHCP Guard policy to VLANs across multiple interfaces:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | **vlan configuration** *vlan_list*<br>**Example:**<br>`Device(config)# vlan configuration 334` | Specifies the VLANs to which the IPv6 Snooping policy will be attached, and enters VLAN interface configuration mode. |
| **Step 4** | **ipv6 dhcp guard** [**attach-policy** *policy_name*]<br>**Example:**<br>`Device(config-vlan-config)# ipv6 dhcp guard attach-policy example_policy` | Attaches the IPv6 Neighbor Discovery policy to the specified VLANs across all switch and stack interfaces. The default policy is attached if the **attach-policy** option is not used. The default policy is, device-role **client**, **no** trusted-port. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-vlan-config)# end` | Exits VLAN interface configuration mode and returns to privileged EXEC mode. |

# Configuring IPv6 Source Guard

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ipv6 source-guard policy** *policy_name*<br>**Example:**<br>`Device(config)# ipv6 source-guard policy example_policy` | Specifies the IPv6 Source Guard policy name and enters IPv6 Source Guard policy configuration mode. |
| **Step 4** | [**deny global-autoconf**] [**permit link-local**] [**default**{. . . }] [**exit**] [**no**{. . . }]<br>**Example:**<br>`Device(config-sisf-sourceguard)# deny global-autoconf` | (Optional) Defines the IPv6 Source Guard policy.<br>• **deny global-autoconf**: Denies data traffic from auto-configured global addresses. This is useful when all global addresses on a link are DHCP-assigned and the administrator wants to block hosts with self-configured addresses to send traffic. |

| | Command or Action | Purpose |
|---|---|---|
| | | • **permit link-local**: Allows all data traffic that is sourced by a link-local address.<br><br>**Note**<br>Trusted option under source guard policy is not supported. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-sisf-sourceguard)# end` | Exits of IPv6 Source Guard policy configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show ipv6 source-guard policy** *policy_name*<br>**Example:**<br>`Device# show ipv6 source-guard policy example_policy` | Shows the policy configuration and all the interfaces where the policy is applied. |

### What to do next

Apply the IPv6 Source Guard policy to an interface.

## Attaching an IPv6 Source Guard Policy to an Interface

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/1/4` | Specifies an interface type and identifier; enters interface configuration mode. |
| **Step 4** | **ipv6 source-guard** [**attach-policy** <*policy_name*> ]<br>**Example:**<br>`Device(config-if)# ipv6 source-guard attach-policy example_policy` | Attaches the IPv6 Source Guard policy to the interface. The default policy is attached if the **attach-policy** option is not used. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show ipv6 source-guard policy** *policy_name*<br>**Example:**<br>`Device#(config)# show ipv6 source-guard policy example_policy` | Shows the policy configuration and all the interfaces where the policy is applied. |

## Attaching an IPv6 Source Guard Policy to a Layer 2 EtherChannel Interface

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface port-channel** *port-channel-number*<br>**Example:**<br>`Device(config)# interface Port-channel 4` | Specifies an interface type and port number and places the switch in the port channel configuration mode. |
| **Step 4** | **ipv6 source-guard** [**attach-policy** *<policy_name>* ]<br>**Example:**<br>`Device(config-if)# ipv6 source-guard attach-policy example_policy` | Attaches the IPv6 Source Guard policy to the interface. The default policy is attached if the **attach-policy** option is not used. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show ipv6 source-guard policy** *policy_name*<br>**Example:**<br>`Device# show ipv6 source-guard policy example_policy` | Shows the policy configuration and all the interfaces where the policy is applied. |

# Configuring IPv6 Prefix Guard

**Note** To allow routing protocol control packets sourced by a link-local address when prefix guard is applied, enable the **permit link-local** command in the source-guard policy configuration mode.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ipv6 source-guard policy** *source-guard-policy*<br>**Example:**<br>`Device(config)# ipv6 source-guard policy my_snooping_policy` | Defines an IPv6 source-guard policy name and enters switch integrated security features source-guard policy configuration mode. |
| **Step 4** | **validate address**<br>**Example:**<br>`Device(config-sisf-sourceguard)# no validate address` | Disables the validate address feature and enables the IPv6 prefix guard feature to be configured. |
| **Step 5** | **validate prefix**<br>**Example:**<br>`Device(config-sisf-sourceguard)# validate prefix` | Enables IPv6 source guard to perform the IPv6 prefix-guard operation. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(config-sisf-sourceguard)# exit` | Exits switch integrated security features source-guard policy configuration mode and returns to privileged EXEC mode. |
| **Step 7** | **show ipv6 source-guard policy** [*source-guard-policy*]<br>**Example:**<br>`Device# show ipv6 source-guard policy policy1` | Displays the IPv6 source-guard policy configuration. |

## Attaching an IPv6 Prefix Guard Policy to an Interface

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/1/4` | Specifies an interface type and identifier, and enters interface configuration mode. |
| **Step 4** | **ipv6 source-guard attach-policy** *policy_name*<br>**Example:**<br>`Device(config-if)# ipv6 source-guard attach-policy example_policy` | Attaches the IPv6 Source Guard policy to the interface. The default policy is attached if the **attach-policy** option is not used. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show ipv6 source-guard policy** *policy_name*<br>**Example:**<br>`Device(config-if)# show ipv6 source-guard policy example_policy` | Shows the policy configuration and all the interfaces where the policy is applied. |

## Attaching an IPv6 Prefix Guard Policy to a Layer 2 EtherChannel Interface

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **interface port-channel** *port-channel-number*<br>**Example:**<br>`Device(config)# interface Port-channel 4` | Specifies an interface type and port number and places the switch in the port channel configuration mode. |
| **Step 4** | **ipv6 source-guard** [**attach-policy** <*policy_name*> ]<br>**Example:**<br>`Device(config-if)# ipv6 source-guard attach-policy example_policy` | Attaches the IPv6 Source Guard policy to the interface. The default policy is attached if the **attach-policy** option is not used. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show ipv6 source-guard policy** *policy_name*<br>**Example:**<br>`Device(config)# show ipv6 source-guard policy example_policy` | Shows the policy configuration and all the interfaces where the policy is applied. |

# Configuring an IPv6 Destination Guard Policy

Beginning in privileged EXEC mode, follow these steps to configure an IPv6 destination guard policy:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ipv6 destination-guard policy** *policy-name*<br>**Example:**<br>`Device(config)# ipv6 destination-guard policy poll` | Defines the destination guard policy name and enters destination-guard configuration mode. |
| **Step 4** | **enforcement** {**always** \| **stressed**}<br>**Example:**<br>`Device(config-destguard)# enforcement always` | Sets the enforcement level for the target address. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **exit**<br>**Example:**<br>`Device(config-destguard)# exit` | Exits destination-guard configuration mode and returns to global configuration mode. |
| **Step 6** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface GigabitEthernet 0/0/1` | Enters interface configuration mode. |
| **Step 7** | **ipv6 destination-guard attach-policy** [*policy-name*]<br>**Example:**<br>`Device(config-if)# ipv6 destination-guard attach-policy pol1` | Attaches a destination guard policy to an interface. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC configuration mode. |
| **Step 9** | **show ipv6 destination-guard policy** [*policy-name*]<br>**Example:**<br>`Device# show ipv6 destination-guard policy pol1` | (Optional) Displays the policy configuration and all interfaces where the policy is applied. |

# Configuration Examples for IPv6 First Hop Security

## Example: Configuring an IPv6 DHCP Guard Policy

### Example of DHCPv6 Guard Configuration

```
Device> enable
Device# configure terminal
Device(config)# ipv6 access-list acl1
Device(config-ipv6-acl)# permit host 2001:DB8:0000:
0000:0000:0000:0000:0001 any
Device(config-ipv6-acl)# exit
Device(config)# ipv6 prefix-list abc permit 2001:0DB8::/64 le 128
Device(config)# ipv6 dhcp guard policy pol1
Device(config-dhcp-guard)# device-role server
Device(config-dhcp-guard)# match server access-list acl1
Device(config-dhcp-guard)# match reply prefix-list abc
Device(config-dhcp-guard)# preference min 0
Device(config-dhcp-guard)# preference max 255
Device(config-dhcp-guard)# trusted-port
Device(config-dhcp-guard)# exit
```

```
Device(config)# interface GigabitEthernet 0/2/0
Device(config-if)# switchport
Device(config-if)# ipv6 dhcp guard attach-policy pol1 vlan add 1
Device(config-if)# exit
Device(config)# vlan 1
Device(config-vlan)# ipv6 dhcp guard attach-policy pol1
Device(config-vlan)# end
```

# Examples: Attaching an IPv6 Source Guard Policy to a Layer 2 EtherChannel Interface

The following example shows how to attach an IPv6 Source Guard Policy to a Layer 2 EtherChannel Interface:

```
Device> enable
Device# configure terminal
Device(config)# ipv6 source-guard policy POL
Device(config-sisf-sourceguard) # validate address
Device(config-sisf-sourceguard)# exit
Device(config)# interface Port-Channel 4
Device(config-if)# ipv6 snooping
Device(config-if)# ipv6 source-guard attach-policy POL
Device(config-if)# end
Device#
```

# Examples: Attaching an IPv6 Prefix Guard Policy to a Layer 2 EtherChannel Interface

The following example shows how to attach an IPv6 Prefix Guard Policy to a Layer 2 EtherChannel Interface:

```
Device> enable
Device# configure terminal
Device(config)# ipv6 source-guard policy POL
Device (config-sisf-sourceguard)# no validate address
Device((config-sisf-sourceguard)# validate prefix
Device(config-sisf-sourceguard)# exit
Device(config)# interface Po4
Device(config-if)# ipv6 snooping
Device(config-if)# ipv6 source-guard attach-policy POL

Device(config-if)# end
```

# Example: Using the Data-Glean Recovery Function

Binding entries can be removed from the binding table for various reasons: the switch may have reset, or you may have used the **clear** commands, and so on. The following example shows how you can use the data-glean recovery function to restore valid binding entries in the binding table.

The scenario used in this example involves interaction between the IPv6 Source Guard, IEEE 802.1x authentication, and SISF-based device-tracking features. Described below is the set-up we are using for this example, along with sample configuration, followed by a description of situations that can cause premature removal of valid entries from the binding table, and finally, the configuration that you must have in-place, for such entries to be restored.

The key aspects of this example set-up are outlined below:

- An IPv6 Source Guard policy is configured and attached to an interface.

  This means that if the source address of an incoming packet is in the binding table, the filter allows the packet into the network. If the address is not in the binding table, entry is denied and the packet entry is dropped. When an entry is removed from the binding table, the filter is also removed, and subsequent packets from that source are dropped.

```
Device# show ipv6 source-guard policy src-guard-policy
Source guard policy src-guard-policy configuration:
  validate address
Policy src-guard-policy is applied on the following targets:
Target               Type  Policy               Feature        Target range
Gi1/0/1              PORT  src-guard-policy     Source guard   vlan all
```

- A custom SISF-based device-tracking policy, which allows gleaning of only DHCP packets and not NDP packets is attached to the same interface as the source guard policy.

  This means that any host in the network can use only a DHCP-assigned IP address to communicate.

```
Device# show device-tracking policy glean_only_DHCP
Device-tracking policy glean_only_DHCP configuration:
  security-level guard
  device-role node
  NOT gleaning from Neighbor Discovery
  gleaning from DHCP6
  NOT gleaning from ARP
  NOT gleaning from DHCP4
  NOT gleaning from protocol unkn
Policy glean_only_DHCP is applied on the following targets:
Target               Type  Policy               Feature        Target range
Gi1/0/1              PORT  glean_only_DHCP      Device-tracking vlan all
```

- IEEE 802.1x authentication is enabled.

  This means only authenticated hosts are allowed to request addresses from the DHCP server and attach themselves to the network.

**Note** The following 802.1x configuration is for example purposes only.

```
<output truncated>

 interface GigabitEthernet 1/0/1
 description 802.1x+MAB+IPT

 authentication control-direction in
 authentication event server dead action authorize vlan <vlan id>
 authentication event no-response action authorize vlan <vlan id>
 authentication event server alive action reinitialize
 authentication host-mode multi-domain
 authentication port-control auto
 authentication periodic
 authentication timer reauthenticate server
 authentication violation protect
 mab
 trust device cisco-phone
 dot1x pae authenticator
 dot1x timeout quiet-period 30
 dot1x timeout server-timeout 5
 dot1x timeout tx-period 1
 dot1x max-req 1
```

```
 dot1x max-reauth-req 1
<output truncated>
```

Events that cause a change in the configuration occur in any typical network. For example, a host may be unplugged from one port and then plugged back into another port, or an interface may flap, or you may have configured the **shutdown**, followed by the **no shutdown** interface configuration commands. For the duration that the host is not connected, or the interface is down, the host or interface is considered "unauthenticated". Because of this absence of host or interface authentication, the corresponding binding table entry is removed from the binding table.

When such a host connects back to the network or when such an interface is restored, the client does not reinstantiate the DHCP sequence until the DHCP lease time expires. Until the DHCP sequence is reinstantiated, a valid address fails to be stored in the binding table. If the entry is not in the binding table, the IPv6 Source Guard's filter function drops all packets initiated by that host.

In order to prevent such a situation, configure the data-glean recovery function.

To configure data-glean recovery, create a custom SISF-based device-tracking policy, configure the data-glean policy parameter to recover binding information from DHCP Server, and attach it to the necessary targets.

**Note** When configuring data-glean recovery from DHCP, for binding information retrieval to work as expected, the DHCPv6 Leasequery configuration (as in RFC 5007), is required. Ensure that the leasequery configuration is enabled on the DHCP Server.

The following sample configuration shows how to add the required "data-glean" policy parameter to the existing custom SISF-based device-tracking policy (`glean_only_DHCP`), to recover binding information. It remains attached to the same target as the IPv6 Source Guard policy, that is, Gigabit Ethernet 1/0/1:

```
Device# configure terminal
Device(config)# device-tracking policy glean_only_DHCP
Device(config-device-tracking)#  data-glean recovery dhcp
Device(config-device-tracking)# exit

Device# show device-tracking policy glean_only_DHCP
Device-tracking policy glean_only_DHCP configuration:
  security-level guard
  device-role node
  data-glean recovery dhcp                      <<< Recovery of binding information is
configured.
  NOT gleaning from Neighbor Discovery
  gleaning from DHCP6
  NOT gleaning from ARP
  NOT gleaning from DHCP4
  NOT gleaning from protocol unkn
Policy glean_only_DHCP is applied on the following targets:
Target               Type  Policy               Feature        Target range
Gi1/0/1              PORT  glean_only_DHCP      Device-tracking vlan all

Device# show device-tracking policies interface Gi1/0/1
Target               Type  Policy               Feature        Target range
Gi1/0/1              PORT  glean_only_DHCP      Device-tracking vlan all
Gi1/0/1              PORT  src-guard-policy     Source guard   vlan all
```

With this additional configuration, valid entries are automatically restored in the binding table if they are removed prematurely.

# Additional References for IPv6 First Hop Security

### Related Documents

| Related Topic | Document Title |
|---|---|
| SISF | Configuring SISF-Based Device Tracking chapter of the *Security Configuration Guide* |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/support |

# Feature History for IPv6 First Hop Security

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | IPv6 First Hop Security | First Hop Security in IPv6 is a set of IPv6 security features, the policies of which can be attached to a physical interface, an EtherChannel interface, or a VLAN. An IPv6 software policy database service stores and accesses these policies. When a policy is configured or modified, the attributes of the policy are stored or updated in the software policy database, then applied as was specified.<br>The IPv6 Snooping Policy feature has been deprecated. Although the commands are visible on the CLI and you can configure them, we recommend that you use the Switch Integrated Security Feature (SISF)-based Device Tracking feature instead.<br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches. |

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Fuji 16.8.1a | IPv6 First Hop Security | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Amsterdam 17.1.1 | IPv6 ND Inspection | Starting with this release, the IPv6 ND Inspection feature is deprecated and the SISF- based device tracking feature replaces it and offers the same capabilities. While the IPv6 ND Inspection commands are still available on the CLI and the existing configuration continues to be supported, the commands will be removed from the CLI in a later release. For more information about the replacement feature, see the *Configuring SISF-Based Device Tracking* chapter in this guide. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 27

# Configuring Switch Integrated Security Features


- Information About SISF, on page 473
- How to Configure SISF, on page 494
- Configuration Examples for SISF, on page 504
- Feature History for SISF, on page 511


# Information About SISF

## Overview

Switch Integrated Security Features (SISF) is a framework developed to optimize security in Layer 2 domains. It merges the IP Device Tracking (IPDT) and *certain* IPv6 first-hop security (FHS) functionality[2], to simplify the migration from IPv4 to IPv6 stack or a dual-stack.

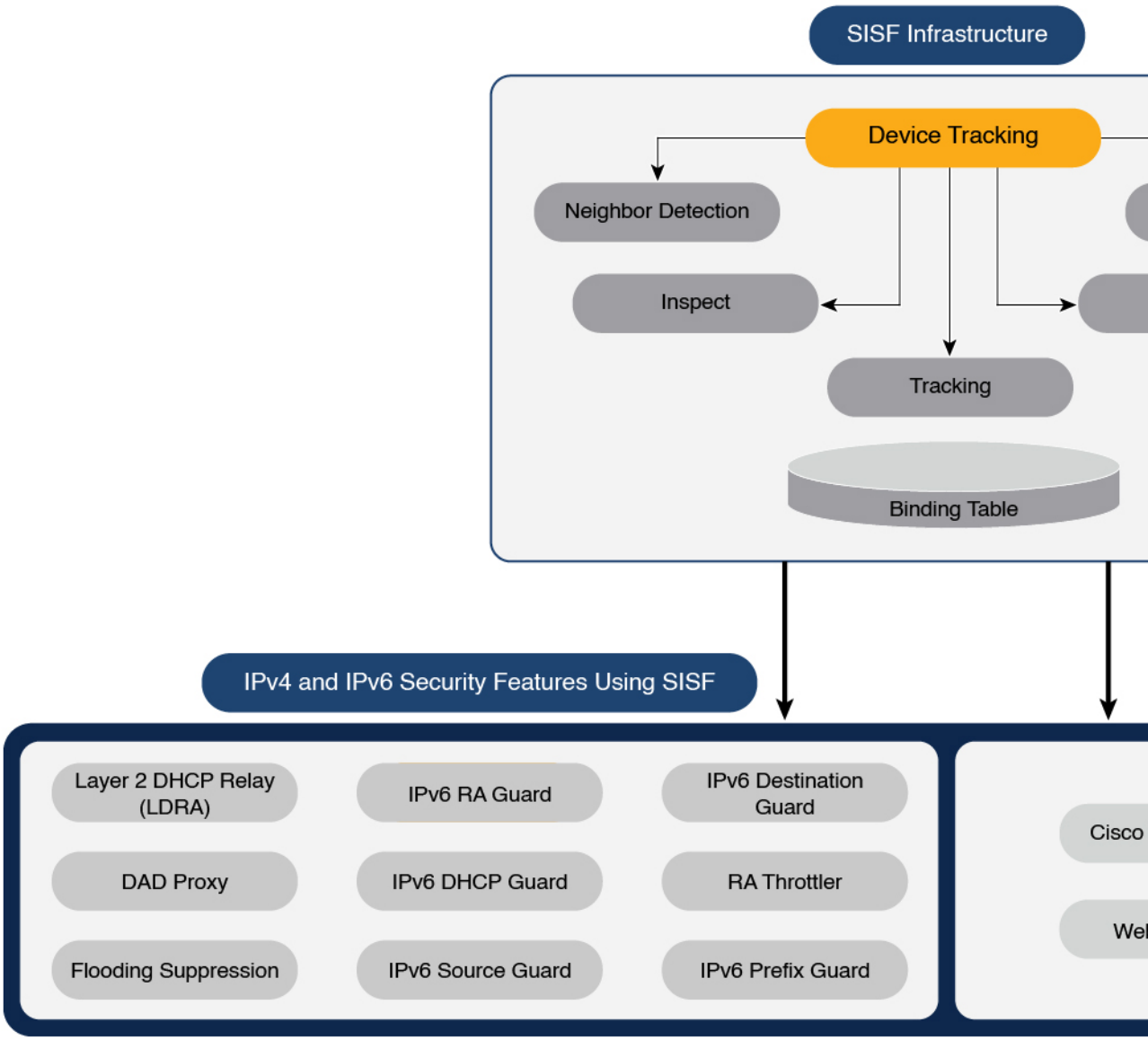
The SISF infrastructure provides a unified database that is used by:

- IPv6 FHS features: IPv6 Router Advertisement (RA) Guard, IPv6 DHCP Guard, Layer 2 DHCP Relay, IPv6 Duplicate Address Detection (DAD) Proxy, Flooding Suppression, IPv6 Source Guard, IPv6 Destination Guard, RA Throttler, and IPv6 Prefix Guard.
- Features like Cisco TrustSec, IEEE 802.1X, Locator ID Separation Protocol (LISP), Ethernet VPN (EVPN), and Web Authentication, which act as clients for SISF.

The following figure illustrates this:

[2] IPv6 Snooping Policy, IPv6 FHS Binding Table Content, and IPv6 Neighbor Discovery Inspection

*Figure 23: SISF Framework*




**Note** The terms “SISF” “device-tracking” and “SISF-based device-tracking” are used interchangeably in this document and refer to the same feature. Neither term is used to mean or should be confused with the legacy IPDT or IPv6 Snooping features.

# Understanding the SISF Infrastructure

This section explains the various elements of the SISF infrastructure as shown in the Figure 23: SISF Framework, on page 474.

## The Binding Table

The SISF infrastructure is built around the binding table. The binding table contains information about the hosts that are connected to the ports of a switch and the IP and MAC address of these hosts. This helps to create a physical map of all the hosts that are connected to a switch.

Each entry in a binding table provides the following information about a connected host:

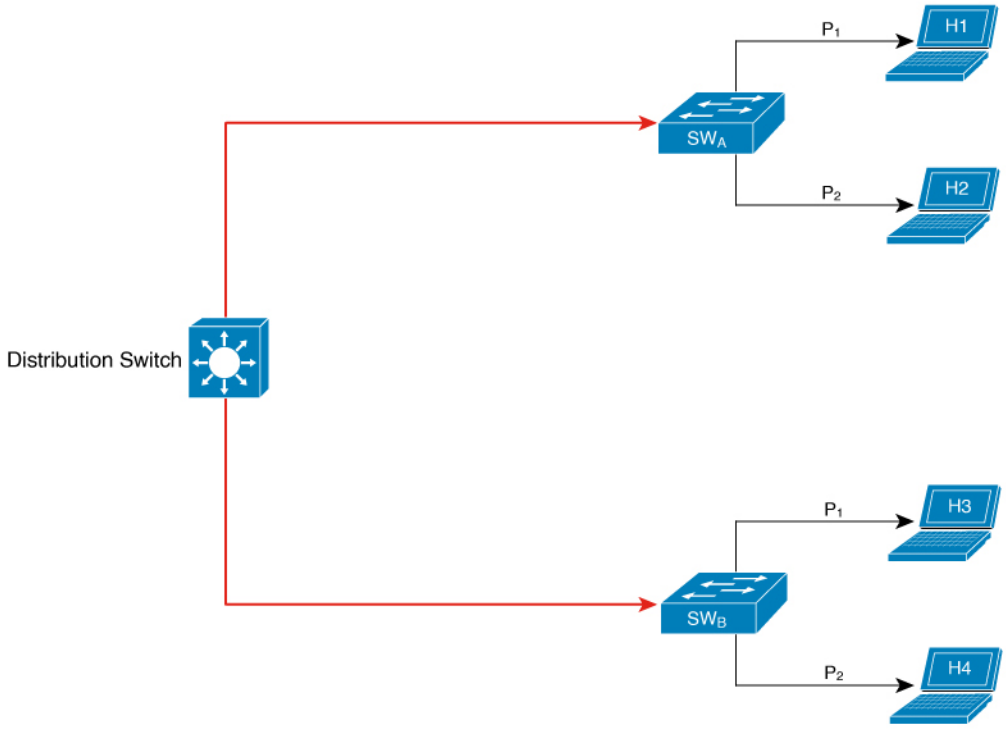
- IPv4 or IPv6 address of the host.
- MAC address of the host. The same MAC address may be linked to an IPv4 and IPv6 address.
- The interface or port on the switch that the host is connected to, and the associated VLAN.
- The state of the entry, which indicates the reachability of the entry.

The following figure shows a simple network topology and a representative binding table for each access switch in the network. $SW_A$ and $SW_B$ are the two access switches in the network. The two access switches are connected to the same distribution switch. H1, H2, H3, H4 are the hosts.

This is an example of a distributed binding table, that is, each access switch in the network has its own table. An alternative set-up could be one centralised binding table on the distribution switch with the entries of $SW_A$ *and* $SW_B$.

Having a distributed or a centralised binding table is a key design choice in the process of implementing SISF in your network and is covered in greater detail in the Understanding Policy Parameters, on page 481 section in this chapter.

***Figure 24: Binding Table***




| VLAN | IPv4/ I |
|---|---|
| 200 | 192.0. |
| 200 | 192.0. |

| VLAN | IPv4/ I |
|---|---|
| 100 | 198.51 |
| 100 | 198.51 |

## States and Lifetime of a Binding Table Entry

The state of an entry indicates if the host is reachable or not. The stable states of a binding table entry are: REACHABLE, DOWN, and STALE. When changing from one state to another, an entry may have other temporary or transitional states such as: VERIFY, INCOMPLETE, and TENTATIVE.

How long an entry remains in a given state is determined by its lifetime and by whether or not the entry is validated successfully. The lifetime of an entry can be policy-driven or configured globally.

To configure the REACHABLE, DOWN, and STALE lifetimes, enter the following command in global configuration mode:

**device-tracking binding** { **reachable-lifetime** { *seconds* | **infinite** } | **stale-lifetime** { *seconds* | **infinite** } | **down-lifetime** { *seconds* | **infinite** } }

### State: Reachable

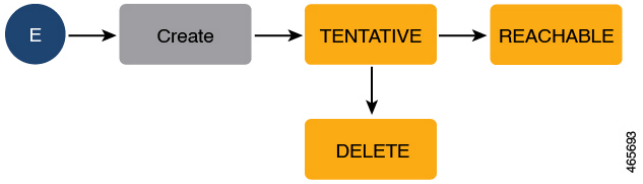
If an entry has this state, it means the host (IP and MAC address) from which a control packet was received, is a verified and valid host. A reachable entry has a default lifetime of 5 minutes. You can also configure a duration. By configuring a reachable-lifetime, you specify how long a host can remain in a REACHABLE state, after the last incoming control packet from that host.

If an event is detected before the entry's reachable lifetime expires, then the reachable lifetime is reset.

To qualify for the REACHABLE state, a new entry goes through the process illustrated in the figure below. The switch detects an event (E), such as an incoming control packet from a connected host and creates an entry. Various events cause the creation of an entry, and these are described in the Binding Table Sources section. The creation of an entry is followed by different transient states, such as TENTATIVE or INCOMPLETE. While in a transitional state, the switch validates and confirms the integrity of the binding entry. If the entry is found to be valid, then the state changes to REACHABLE.

But if an address theft or similar event is detected, then the entry is regarded as invalid and is deleted. For example, if an attacker sends unsolicited neighbor advertisement messages with the same IP as the target IP and its (attacker's) own MAC address to redirect traffic.

***Figure 25: Creation of a Reachable Entry***

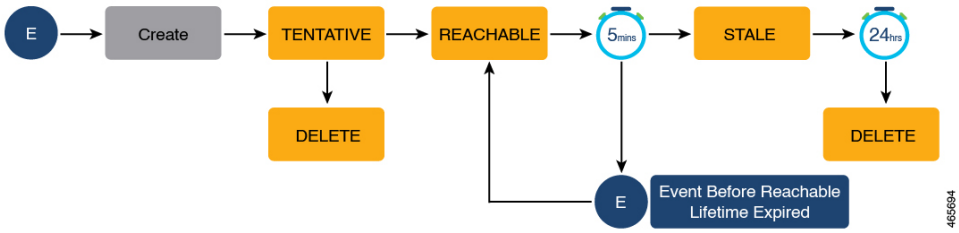



### State: Stale

If an entry is in this state it means that the entry's reachable lifetime has expired and the corresponding host is still silent (no incoming packets from the host). A stale entry has a default lifetime of 24 hours. You can also configure a duration. An entry that remains in the STALE state beyond the stale lifetime, is deleted.

This is illustrated in the figure below which depicts the lifecycle of an entry.

*Figure 26: Lifecycle of an Entry*




### State: Down

If an entry is in this state, it means that the host's connecting interface is down. A down entry has a default lifetime of 24 hours. You can also configure a duration. An entry that remains in the DOWN state beyond the down lifetime, is deleted.

### Polling a Host and Updating the Binding Table Entry

Polling is a periodic and conditional checking of the host to see the state it is in, whether it is still connected, and whether it is communicating. In addition to determining an entry's state, you can use polling to reconfirm an entry's state.

You can enable polling with the **device-tracking tracking** command in global configuration mode. After you do, you still have the flexibility to turn polling on or off for a particular interface or VLAN. For this, configure the **tracking enable** or **tracking disable** keywords in the policy (the device-tracking configuration mode). When polling is enabled, the switch polls the host at the specified interval, thus reconfirming its reachability for the duration of its reachable lifetime.

When polling is enabled, the switch sends up to three polling requests, after the reachable lifetime expires, at system-determined intervals. You can also configure this interval with the **device-tracking tracking retry-interval** *seconds* command in global configuration mode.

The figure below depicts the lifecycle of an entry where the host is polled. Default reachable and stale lifetimes, and retry intervals are used in figure:

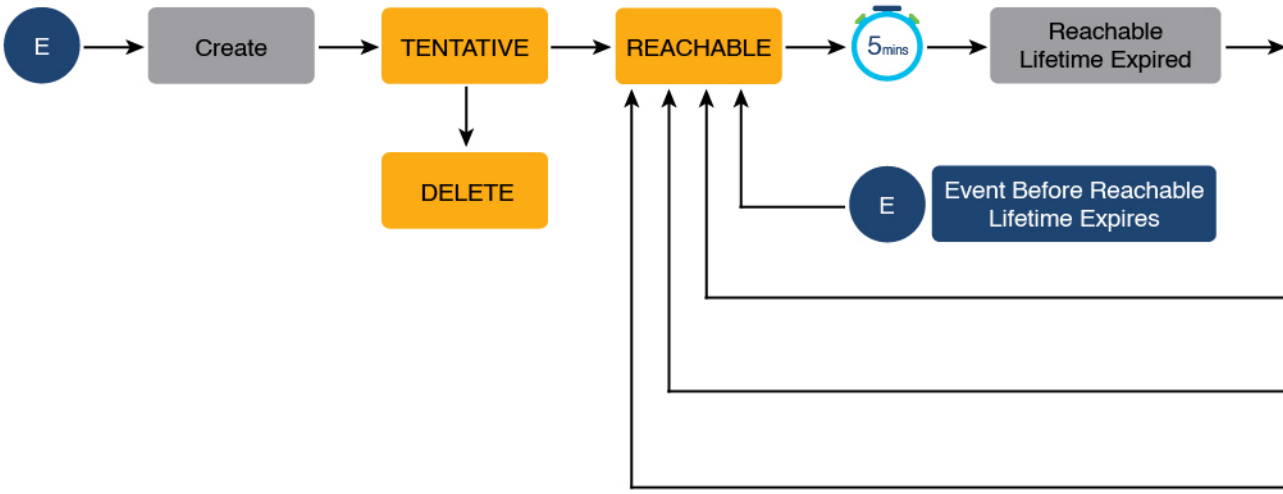
An event (E) is detected and a REACHABLE entry is created.

If an event is detected *during* the reachable lifetime, the reachable lifetime timer is reset.

The switch sends a polling request after the reachable lifetime expires. The switch polls the host up to three times at fixed, system-determined intervals. The polling request may be in the form of a unicast Address Resolution Protocol (ARP) probe or a Neighbor Solicitation message. During this time the state of the entry changes to VERIFY. If a polling response is received (thus confirming reachability of the host), the state of the entry changes back to REACHABLE.

If the switch does not receive a polling response after three attempts, the entry changes to the STALE state. It remains in this state for 24 hours. If an event is detected during the stale lifetime, the state of the entry is changed back to REACHABLE. At expiry of the stale lifetime, the device sends one final polling to ascertain reachability. If this final polling attempt receives a reply, the state of the entry is changed back to REACHABLE. If the final polling attempt does not receive a response, the entry is deleted.

*Figure 27: Lifecycle of an Entry Where the Host is Polled*




## Binding Table Sources

The following are the sources of information and events that cause the creation and update of a binding table entry:

- Learning events that dynamically populate the binding table:
  - Dynamic Host Configuration Protocol (DHCP) negotiation (DHCP REQUEST, and DHCP REPLY). This includes DHCPv4 and DHCPv6.
  - Address Resolution Protocol (ARP) packets.

    Starting with Cisco IOS XE Amsterdam 17.3.1, ARP packets are throttled to mitigate high CPU utilization scenarios. In a five second window, a maximum of 50 ARP packets, from the same source, are processed by SISF. Note that the limit of 50 in five seconds is for each binding entry, that is, for each source IP.

    All ARP (ARP REQUEST and ARP REPLY) packets are dropped if
    - the limit is reached,
    - Cisco Software-Defined Access is configured, and
    - the security level of the device tracking policy is set to guard.
  - Neighbor Discovery Protocol (NDP) packets.

- Multiple Identity Association-Nontemporary Address (IA_NA) and Identity Association-Prefix Delegation (IA_PD).

  In some cases, a network device can request and receive more than one IPv6 address from the DHCP server. This may be done to provide addresses to multiple clients of the device, such as when a residential gateway requests addresses to distribute to its LAN clients. When the device sends out a DHCPv6 packet, the packet includes all of the addresses that have been assigned to the device.

  When SISF analyzes a DHCPv6 packet, it examines the IA_NA (Identity Association-Nontemporary Address) and IA_PD (Identity Association-Prefix Delegation) components of the packet and extracts each IPv6 address contained in the packet. SISF adds each extracted address to the binding table.

Entries created through learning events like the ones listed above are called "dynamic entries". In the output of the **show device-tracking database details** privileged EXEC command, such entries are prefixed with an abbreviation that clarifies the kind of dynamic learning event it was. For example, ARP for ARP packets, ND for NDP packets, and so on.

- Configuring static binding entries.

  If there are silent but reachable hosts in the Layer 2 domain, you can create static binding entries to retain binding information even if the host becomes silent.

  A static binding entry is a binding entry that is manually added to the binding table, by configuring the following command in global configuration mode:

  **device-tracking binding vlan** *vlan_id* { *ipv4_add ipv6_add ipv6_prefix* } [ **interface** *inteface_type_no* ] [ *48-bit-hardware-address* ] [ **reachable-lifetime** { *seconds* | **default** | **infinite** } | **tracking** { **default** | **disable** | **enable** [ **retry-interval** { *seconds* | **default** } ] } [ **reachable-lifetime** { *seconds* | **default** | **infinite** } ] ]

  In the output of the **show device-tracking database details** privileged EXEC command, static entries are prefixed with the letter "S".

  You can configure a reachable lifetime for a static entry. The stale and down lifetime timer is fixed by the system as **infinite** (For an entry in the STALE or DOWN state, the output of the **show device-tracking database** command displays the `Time Left` column as “N/A”). This means that when a static entry enters the STALE or DOWN state it remains in this state, and in the binding table, indefinitely.

  A static entry can be removed from the binding table only by the actions listed below. It cannot be deleted from the binding table by using **clear** commands or by any other event:

  - You remove the entry by configuring the **no** form of the above command.
  - A local entry replaces the static entry.

    A local entry is an entry that is automatically created by the system when you configure an SVI on the device. When configuring the SVI, if you use the same IP address as the static entry then the static entry is replaced with the local entry, because the local entry has a higher priority.

    This replacement of a static entry by a local entry is introduced only in Cisco IOS XE Bengaluru 17.4.1 onwards; it does not happen in earlier releases.

    In the output of the **show device-tracking database details** privileged EXEC command, local entries are prefixed with the letter "L".

For more information about static binding entries, see the **device-tracking binding** command in the command reference.

**Note** In addition to the primary or key events listed above, there is a specific scenario in which a ping can result in a device-tracking entry. If a sender's ARP cache or IPv6 neighbor table doesn't have the target's IP address yet, then a ping triggers an ARP packet for IPv4, or ND packet for IPv6. This can result in a device-tracking entry.

But if the target IP is already in the ARP cache or IPv6 neighbour table, no ARP or ND packet is generated when you ping - in which case SISF cannot learn the IP address.

## Device-Tracking

SISF-based device-tracking is disabled by default. You can enable the feature on an interface or VLAN.

When you enable the feature, the binding table is created, followed by subsequent maintenance of the binding table.

The events listed in the Binding Table Sources, on page 478 section act as triggers for SISF-based device-tracking, to track the presence, location, and movement of hosts in the network, to populate and maintain the binding table. For example, if information about a host is learnt by means of an ARP or ND packet, every subsequent ARP or ND packet from the same host acts as an alert for SISF-based device-tracking, to refresh the entry in the binding table, thus indicating if the host is still present in the same location or has moved.

The continuous process of snooping of packets that the switch receives, extraction of device identity (MAC and IP address), and storage of information in the binding table of the switch, ensures binding integrity and maintains the reachability status of the hosts in the binding table.

For information how to enable SISF-based device-tracking, see How to Configure SISF, on page 494.

## Actions on a Packet and Actions on the Binding Table

The distinction between system behaviour in the context of a binding table entry and system behaviour in the context of a packet (from which the binding information is extracted) is an important one, because the available actions are exclusive to each context.

**SISF actions on a packet** determine if any other feature is allowed to access, or use, or forward the packet.

- Stop: Means the packet is not available to any client or feature.

  A packet may be stopped while the binding integrity of a possible entry is being verified. From a system perspective,this action is equivalent to a packet drop, but from a SISF perspective, this is not necessarily seen as a malicious packet (as in case of drop), as SISF tries to glean the binding information from it.

- Forward: Means the packet is allowed to enter the network and is sent on, unchanged.

- Drop: Means the packet is not allowed to enter the network.

  Until Cisco IOS XE Bengaluru 17.6.x, the drop action is restricted to ND Packets.

**SISF actions on the binding table** include only the following:

- Create or update: Means the packet or other source is used to create or update an entry in the binding table.

- Ignore: Means the packet or other source of information is disregarded and there is no change or update in the binding table.

Multiple actions may be performed on a packet: If the *stop* and *ignore* actions are observed for a packet, it may also result in a *drop*. If *stop* and *update* actions are observed, the packet is allowed to enter the network and is sent on, unchanged.

## Device-Tracking Policy

A device-tracking policy is a set of rules that SISF-based device-tracking follows. The policy dictates which events will be listened to, whether a host will be probed, the wait time before the host is probed, and so on. These rules are referred to as policy parameters.

**Note** The policy must be attached to an interface or VLAN. Only then is the binding table for that interface or VLAN populated - in accordance with policy parameters.

For information about the various ways in which you can create a policy, see How to Configure SISF, on page 494.

To display a policy's settings, use the **show device-tracking policy** *policy_name* command in privileged EXEC mode.

# Understanding Policy Parameters

Policy parameters are the keywords available for configuration in the device-tracking configuration mode. Each policy parameter addresses one or more aspects of network security.

This section explains the purpose of *some* of the important policy parameters so you can configure your policy to better suit your requirements.

```
Device(config)# device-tracking policy example_policy
Device(config-device-tracking)# ?
device-tracking policy configuration mode:

  device-role           Sets the role of the device attached to the port
  limit                 Specifies a limit
  security-level        setup security level
  tracking              Override default tracking behavior
  trusted-port          setup trusted port
```

For information about all the paramters displayed in the device-tracking configuration mode, see the command reference document of the corresponding release.

## Glean versus Guard versus Inspect

When a packet enters the network, SISF extracts the IP and MAC address (the source of the packet) and subsequent action, is dictated by the security-level that is configured in the policy.

Glean, guard, and inspect are the options available under the security-level parameter. Glean is the least secure option, inspect, is moderately secure, and guard, is the most secure.

To configure this parameter in a policy, enter the **security-level** keyword in the device-tracking configuration mode.

### Glean

When the security-level is set to **glean**, SISF extracts the IP and MAC address and enters them into the binding table, without any verification. This option therefore does not ensure binding integrity. It may for example, be suited to a set-up where client applications such as IEEE 802.1X or SANET want to only learn about the host and not rely on SISF for authentication.

The only factor that affects the addition of the binding entry for this security-level, is the address count limit. There are separate limits for the maximum number of IPs per port, IPv4 per MAC, and IPv6 per MAC. Entries are rejected once a limit is reached. For more information about this parameter, see Address Count Limits, on page 491.

### Guard

This is the default value for the security-level parameter.

When the security-level is set to **guard**, SISF extracts and verifies the IP and MAC address of packets entering the network. The outcome of the verification determines if a binding entry is added, or updated, or if the packet is dropped and the client is rejected.

The process of verification starts with the search for a matching entry in the database. The database may be centralised or distributed. If a matching entry is not found, a new entry is added.

If a matching entry is found and the points of attachment (MAC, VLAN, or interface) are found to be the same, only the timestamp is updated. If not, the scope of verification is extended to include validation of address ownership. This may include host polling to determine if the change in the point of attachment (a different MAC, or VLAN) is valid. If the change is valid the entry is updated, or if it is a case of theft, the entry is not added to the binding table.

If a binding entry is added or updated, the corresponding client is granted access to the network. If an entry does not pass verification, the corresponding client is rejected.

**Note** The verification process affects the binding entry and the corresponding incoming packet.

### Inspect

Even though security-level **inspect** is available on the CLI, we recommend not using it. The **glean** and **guard** options described above address most use cases and network requirements.

### Security Level and SISF Action on a Packet

The security level policy parameter can affect actions on ND and ARP packets: If the security level is **guard** and the packet doesn't pass verification, a drop action follows. If its glean, the packet is not dropped. Verification failure (with the **guard** security level) can be because of various reasons including invalid binding information, reaching the address count limit, and so on.

Note that if the packet is an ARP packet, the *drop* action itself is not possible until Cisco IOS XE Cupertino 17.7.1.

## Trusted-Port and Device-Role Switch

The **device-role switch** and **trusted-port** options help you design an efficient and scalable "secure zone". When used together, these two parameters help you achieve an efficient distribution of the creation of entries in the binding table. This keeps the binding tables size under control.

The **trusted-port** option: Disables the guard function on configured targets. Bindings learned through a trusted-port have preference over bindings learned through any other port. A trusted port is also given preference in case of a collision while making an entry in the table.

The **device-role** option: Indicates the type of device that is facing the port and this can be a node or a switch. To allow the creation of binding entries for a port, you configure the device as a node. To stop the creation of binding entries, you configure the device as switch.

Configuring the device as a switch is suited to multi-switch set-ups, where the possibility of large device tracking tables is very high. Here, a port facing a device (an uplink trunk port) can be configured to stop creating binding entries, and the traffic arriving at such a port can be trusted, because the switch on the other side of the trunk port will have device-tracking enabled and that will have checked the validity of the binding entry.

**Note** While there are scenarios where configuring only either one of these options may be suitable, the more common use case is for both the **trusted-port** and **device-role switch** options to be configured on the port - the examples below explain this in detail. Possible scenarios where only either one of these options is suited or required have also been described, at the end of this section.

To configure these parameters in a policy, enter the **trusted-port** and **device-role** keywords in the device-tracking configuration mode.

#### Example: Using Trusted-Port and Device-Role Switch Options in a Multi-Switch Set-Up

The following example explains how the **device-role switch** and **trusted-port** options help to design an efficient and scalable "secure zone".

In figure Figure 28: Multi-Switch Set-Ups Without Trusted-Port and Device-Role Switch Options, on page 484 below, $SW_A$, $SW_B$, and $SW_C$ are three access switches. They are all connected to a common distribution switch. The only required configuration on the distribution switch in this scenario is to ensure that traffic of any kind is *not* blocked.

H1, H2, …H6 are the hosts. Each switch has two directly connected hosts. All hosts are communicating with each other, that is, control packets are being transmitted. All hosts are also within the same VLAN boundary. Each switch is receiving control packets from hosts that are directly connected to it, and also from hosts that are connected to other switches. This means $SW_A$ is receiving control packets from H1, H2, …H6 similarly with $SW_B$ and $SW_C$.

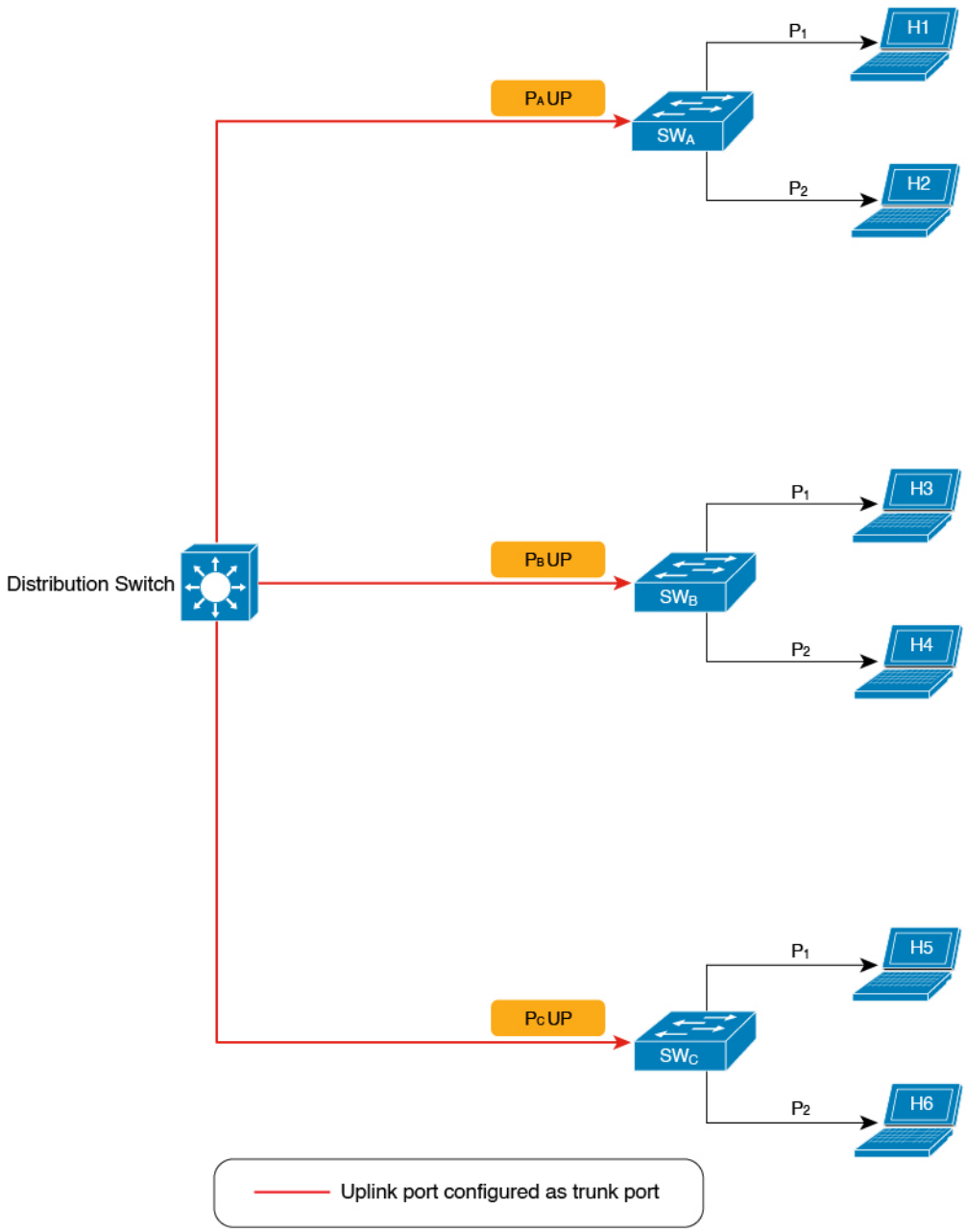
For each switch, the entries of directly connected hosts have interface or port $P_1$ and $P_2$ in the binding table. Entries originating from hosts that are connected to other switches have interface or port name $P_x$UP, to show that they have been learned through the uplink port (x represents the corresponding uplink port for each switch. For example, the entries that $SW_A$ has learnt through its uplink port have interface or port name $P_A$UP and for $SW_B$ it is $P_B$UP, and so forth.

The end result is that each switch learns and creates binding entries for all hosts in the set-up.

This scenario displays an inefficient use of the binding table, because each host is being validated multiple times, which does not make it more secure than if just one switch validates host. Secondly, entries for the

same host in multiple binding tables could mean that the address count limit is reached sooner. After the limit is reached, any further entries are rejected and required entries may be missed this way.

***Figure 28: Multi-Switch Set-Ups Without Trusted-Port and Device-Role Switch Options***




| VLAN | IPv4/ IPv6 |
|---|---|
| 100 | 192.0.2.1 |
| 100 | 192.0.2.2 |
| 200 | 192.0.2.3 |
| 200 | 192.0.2.4 |
| 300 | 192.0.2.5 |
| 300 | 192.0.2.6 |

| VLAN | IPv4/ IPv6 |
|---|---|
| 200 | 192.0.2.3 |
| 200 | 192.0.2.4 |
| 100 | 192.0.2.1 |
| 100 | 192.0.2.2 |
| 300 | 192.0.2.5 |
| 300 | 192.0.2.6 |

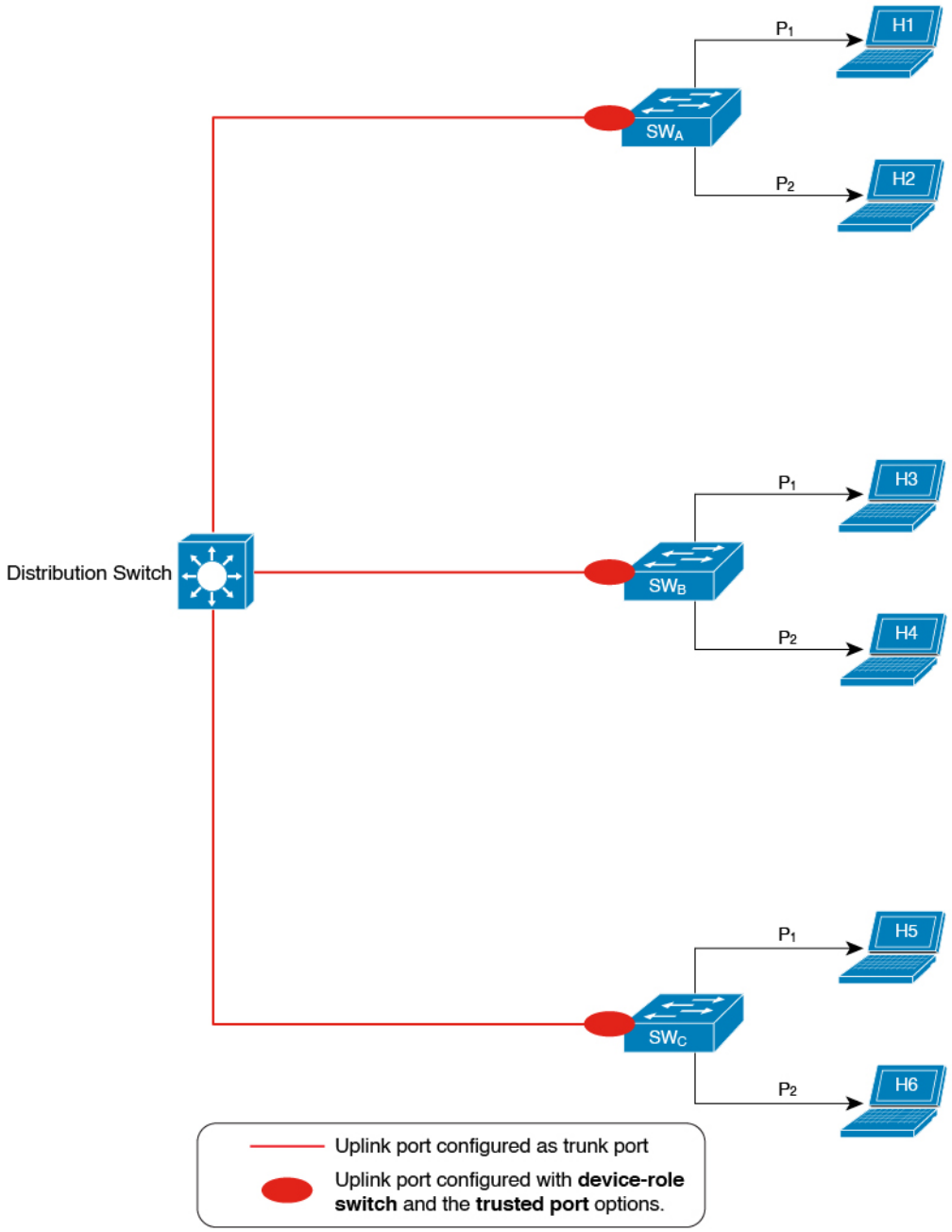
| VLAN | IPv4/ IPv6 |
|---|---|
| 300 | 192.0.2.5 |
| 300 | 192.0.2.6 |
| 100 | 192.0.2.1 |
| 100 | 192.0.2.2 |
| 200 | 192.0.2.3 |
| 200 | 192.0.2.4 |

By contrast see figure Figure 29: Multi-Switch Set-Ups With Trusted-Port and Device-Role Switch Options, on page 486 below. Here when $SW_A$ intercepts the packet of a host that is not attached to it (say H3 which is directly attached to $SW_B$), it does not create an entry because it detects that H3 is attached to a device that is configured as a switch (**device-role switch** option) and the uplink port of the switch (where the packet came from) is a trusted port (**trusted-port** option).

By creating binding entries only on switches where the host appears on an access port (port $P_1$ and $P_2$ of each switch), and not creating entries for a host that appears over an uplink port or trusted port ($P_x$ UP), each switch in the set-up validates and makes only the required entries, thus achieving an efficient distribution of the creation of binding table entries.

A second advantage of configuring **device-role switch** and **trusted-port** options in a multi-switch scenario is that it prevents duplicate entries when a host, say H1 moves from one switch to another. H1's IP and MAC binding in the earlier location (let's say $SW_A$) continues to remain there until it reaches the STALE state. But if H1 moves and connects to a second switch, say $SW_C$, then $SW_A$ receives a duplicate binding entry through the uplink port. In such a situation, if the uplink port of the second switch ($SW_C$) is configured as a trusted port, $SW_A$ deletes its stale entry. Further, it doesn't create another new binding entry because the $SW_C$ will already have the latest entry and this entry is trusted.

*Figure 29: Multi-Switch Set-Ups With Trusted-Port and Device-Role Switch Options*



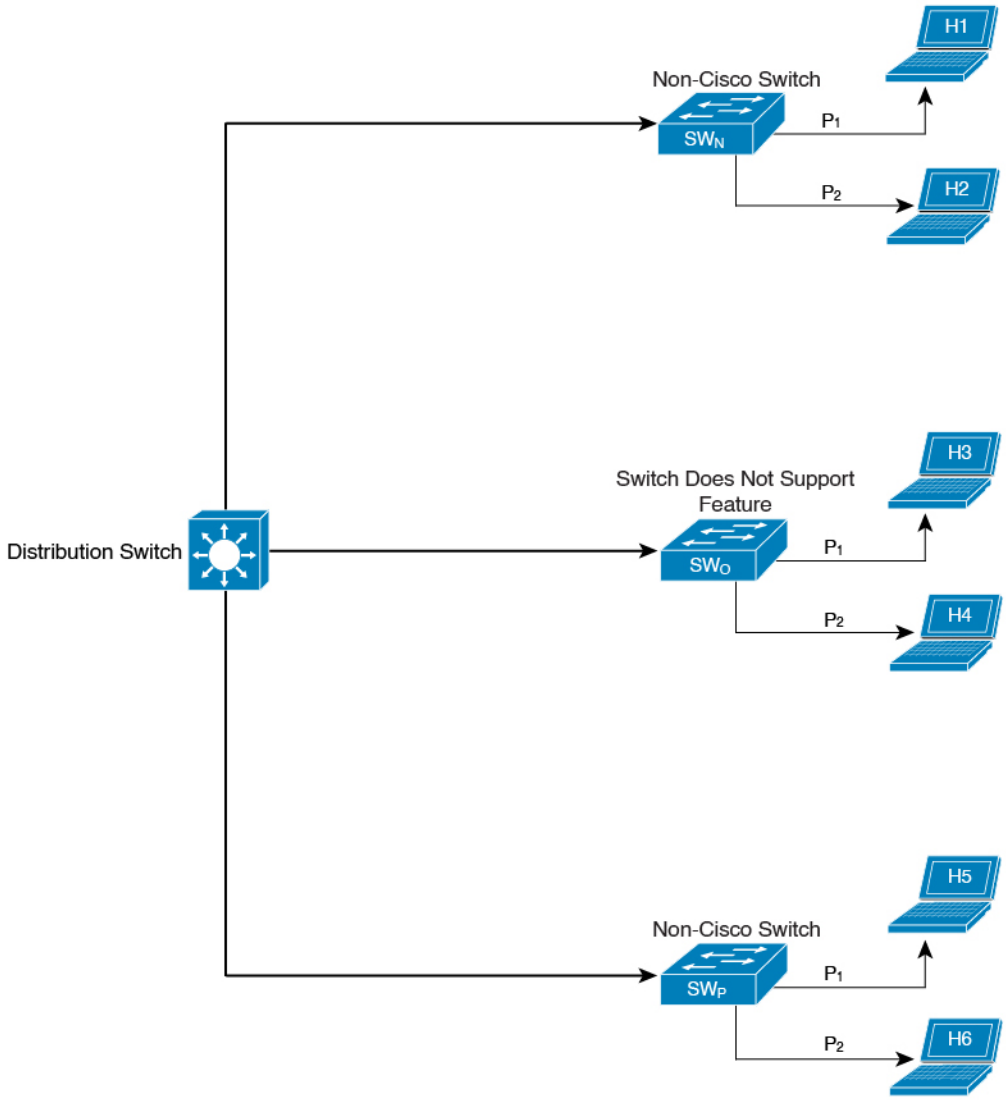

### Example: When Not to Use Trusted-Port and Device-Role Switch Options

While the previous example clarifies how a multi-switch set-up with distributed binding tables stands to benefit from the **device-role switch** and **trusted-port** options, it may not suit networks of the following kinds:

- Networks where non-Cisco switches are being used
- Networks where the switch does not support the SISF-based device-tracking feature.

In both cases, we recommended that you not configure the **device-role switch** and **trusted-port** options. Further, we recommended that you maintain a centralised binding table - on the distribution switch. When you do, all the binding entries for all the hosts connected to non-Cisco switches and switches that do not support the feature, are validated by the distribution switch and still secure your network. The figure below illustrates the same.

***Figure 30: Centralised Binding Table***




| VLAN | IPv4/ I |
|---|---|
| 100 | 192.0. |
| 100 | 192.0. |
| 200 | 192.0. |
| 200 | 192.0. |
| 300 | 192.0. |
| 300 | 192.0. |

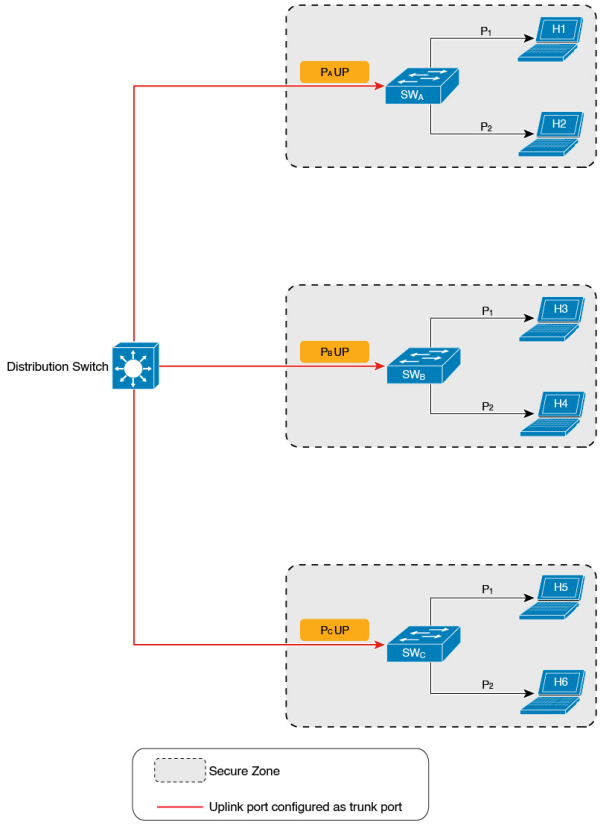
## Creating an Efficient and Scalable Secure Zone

By using the **trusted-port** and **device-role switch** options in suitable networks and leaving them out in others, you can achieve an efficient and scalable secure zone.

Secure Zones 1, 2 and 3, display three different set-ups and the secure zone that is established in each case.

| **Secure Zone:** | **Secure Zone 1 - Inefficient and Unscalable Secure Zone** | **Secure Zone 2 - Efficient and Scalable Secure Zone When Binding Tables are Decentralized** | **Secure Zone 3: Efficient Secure Zone When Binding Table is Centralized** |
|---|---|---|---|
| **Scalability:** | Unscalable; each switch has entries of all the hosts in the network | Scalable; each switch as entries of only directly connected hosts | Unscalable; the distribution switch has entries of all hosts in the network |
| **Polling and its effect on the network:**<br>n = number of hosts<br>m = number of switches<br>total number of polling requests: = n X m | 18 polling requests are being sent (6 hosts x 3 switches).<br>Each host is polled by all the switches in the network (in the absence of the **trusted-port** and **device-role switch** options).<br>Network load is very high. | 6 polling requests are being sent (2 hosts x 1 switch for *each* switch).<br>Minimal network load. (Polling requests are sent by the local access switches to directly connected hosts, each polling request passes through fewer points in the network.) | 6 polling requests are being sent (6 hosts x 1 switch)<br>Network load is higher than secure zone 2, but not as high as secure zone 1. (Polling requests come from the distribution switch and go through the access switch before reaching the host.) |
| **Efficiency:** | Inefficient binding table, because the binding table is duplicated on each switch. | Efficient binding table, because each host's binding information is entered only once, and in one binding table and this the binding table of the directly connected switch. | Efficient binding table, because the binding information for each host is entered only once, and this is in the central binding table, which is on the distribution switch. |
| **Recommended Action:** | Reapply suitable policies to make the secure zone like secure zone 2 | None; this is an efficient and scalable secure zone. | None; this is the best possible secure zone given the type of set-up (where the other switches in the network are either non-Cisco or do not support the feature) |

*Figure 31: Secure Zone 1 - Inefficient and Unscalable Secure Zone*




Binding table for $SW_A$

| VLAN | IPv4/ IPv6 Address | MAC Address | Interface or Port | State | |
|---|---|---|---|---|---|
| 100 | 192.0.2.1 | 00:1A:C2:7B:00:47 | $P_1$ | REACHABLE | H1 |
| 100 | 192.0.2.2 | 00:1A:C2:7B:00:47 | $P_2$ | REACHABLE | H2 |
| 200 | 192.0.2.3 | 00:1A:C2:7B:00:48 | $P_A$UP | REACHABLE | H3 |
| 200 | 192.0.2.4 | 00:1A:C2:7B:00:48 | $P_A$UP | REACHABLE | H4 |
| 300 | 192.0.2.5 | 00:1A:C2:7B:00:49 | $P_A$UP | REACHABLE | H5 |
| 300 | 192.0.2.6 | 00:1A:C2:7B:00:49 | $P_A$UP | REACHABLE | H6 |

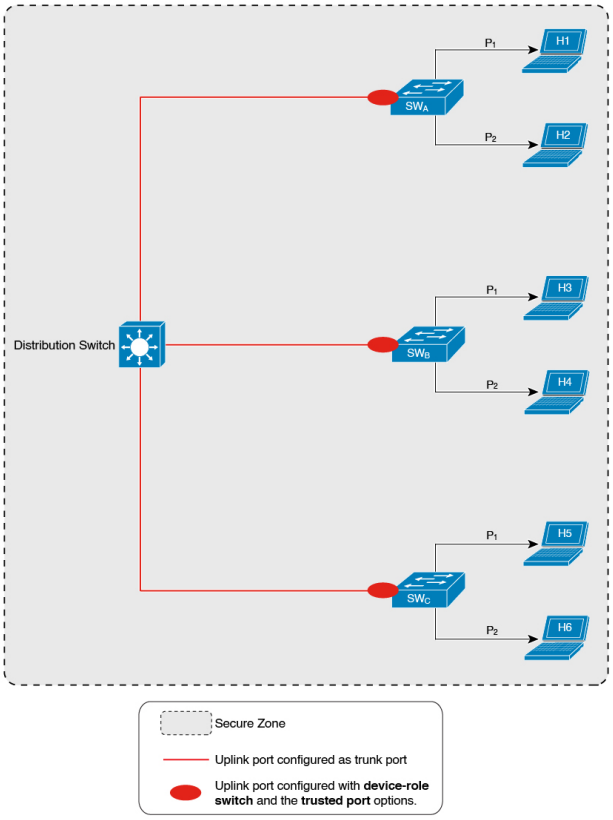
Binding table for $SW_B$

| VLAN | IPv4/ IPv6 Address | MAC Address | Interface or Port | State | |
|---|---|---|---|---|---|
| 200 | 192.0.2.3 | 00:1A:C2:7B:00:48 | $P_1$ | REACHABLE | H3 |
| 200 | 192.0.2.4 | 00:1A:C2:7B:00:48 | $P_2$ | REACHABLE | H4 |
| 100 | 192.0.2.1 | 00:1A:C2:7B:00:47 | $P_B$UP | REACHABLE | H1 |
| 100 | 192.0.2.2 | 00:1A:C2:7B:00:47 | $P_B$UP | REACHABLE | H2 |
| 300 | 192.0.2.5 | 00:1A:C2:7B:00:49 | $P_B$UP | REACHABLE | H5 |
| 300 | 192.0.2.6 | 00:1A:C2:7B:00:49 | $P_B$UP | REACHABLE | H6 |

Binding table for $SW_C$

| VLAN | IPv4/ IPv6 Address | MAC Address | Interface or Port | State | |
|---|---|---|---|---|---|
| 300 | 192.0.2.5 | 00:1A:C2:7B:00:49 | $P_1$ | REACHABLE | H5 |
| 300 | 192.0.2.6 | 00:1A:C2:7B:00:49 | $P_2$ | REACHABLE | H6 |
| 100 | 192.0.2.1 | 00:1A:C2:7B:00:47 | $P_C$UP | REACHABLE | H1 |
| 100 | 192.0.2.2 | 00:1A:C2:7B:00:47 | $P_C$UP | REACHABLE | H2 |
| 200 | 192.0.2.3 | 00:1A:C2:7B:00:48 | $P_C$UP | REACHABLE | H3 |
| 200 | 192.0.2.4 | 00:1A:C2:7B:00:48 | $P_C$UP | REACHABLE | H4 |

*Figure 32: Secure Zone 2 - Efficient and Scalable Secure Zone When Binding Tables are Decentralized*




Binding table for $SW_A$

| VLAN | IPv4/ IPv6 Address | MAC Address | Interface or Port | State | |
|---|---|---|---|---|---|
| 100 | 192.0.2.1 | 00:1A:C2:7B:00:47 | $P_1$ | REACHABLE | H1 |
| 100 | 192.0.2.2 | 00:1A:C2:7B:00:47 | $P_2$ | **REACHABLE** | H2 |

Binding table for $SW_B$

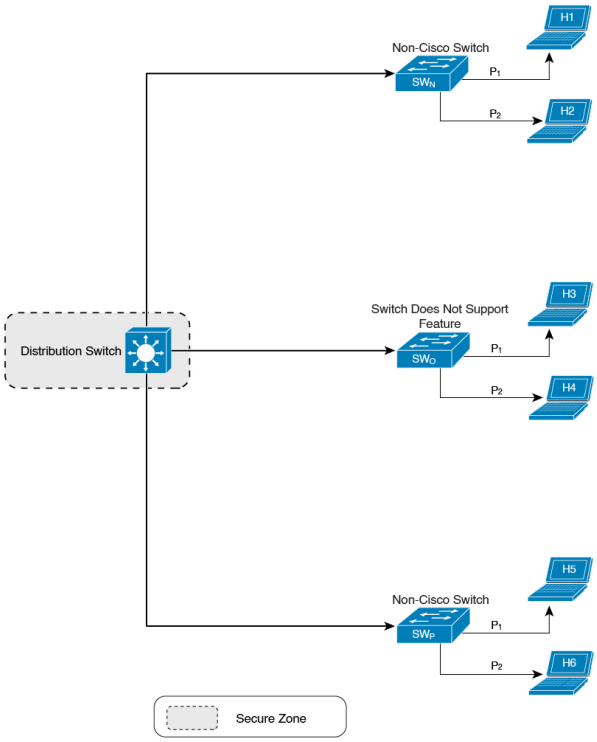
| VLAN | IPv4/ IPv6 Address | MAC Address | Interface or Port | State | |
|---|---|---|---|---|---|
| 200 | 192.0.2.3 | 00:1A:C2:7B:00:48 | $P_1$ | REACHABLE | H3 |
| 200 | 192.0.2.4 | 00:1A:C2:7B:00:48 | $P_2$ | REACHABLE | H4 |

Binding table for $SW_C$

| VLAN | IPv4/ IPv6 Address | MAC Address | Interface or Port | State | |
|---|---|---|---|---|---|
| 300 | 192.0.2.5 | 00:1A:C2:7B:00:49 | $P_1$ | REACHABLE | H5 |
| 300 | 192.0.2.6 | 00:1A:C2:7B:00:49 | $P_2$ | REACHABLE | H6 |

465701

*Figure 33: Secure Zone 3: Efficient Secure Zone When Binding Table is Centralized*




Binding table for Distribution Switch

| VLAN | IPv4/ IPv6 Address | MAC Address | Interface or Port | State | |
|---|---|---|---|---|---|
| 100 | 192.0.2.1 | 00:1A:C2:7B:00:47 | $P_N$UP | REACHABLE | H1 |
| 100 | 192.0.2.2 | 00:1A:C2:7B:00:47 | $P_N$UP | REACHABLE | H2 |
| 200 | 192.0.2.3 | 00:1A:C2:7B:00:48 | $P_O$UP | REACHABLE | H3 |
| 200 | 192.0.2.4 | 00:1A:C2:7B:00:48 | $P_O$UP | REACHABLE | H4 |
| 300 | 192.0.2.5 | 00:1A:C2:7B:00:49 | $P_P$UP | REACHABLE | H5 |
| 300 | 192.0.2.6 | 00:1A:C2:7B:00:49 | $P_P$UP | REACHABLE | H6 |

### When to Use Only Trusted-Port or Only Device-Role Switch

Configuring only **device-role switch** is suited to situations when you want to listen but not learn entries. For example, for Duplicate Address Detection (DAD), or when you want to send IPv6 or Neighbor Solicitation (NS) message on a switch-facing port.

When you configure this option on a switch port (or interface), SISF-based device-tracking treats the port as a trunk port, implying that the port is connected to other switches. It does not matter whether the port is actually a trunk port or not. Therefore, when NS packets or queries are sent to switches in the network for new entry validation, only the secure ports (ports where the **device-role switch** is configured) receive the packet or query. This safeguards the network. If the command is not configured on any port, a general broadcast of the query is sent.

Configuring only **trusted-port** is suited to situations where an access port should be configured as a trusted port. If an access port is connected to a DHCP server or a similar service that the switch is consuming, configuring an access port as a trusted port ensures that the service is not disrupted because traffic from such a port is trusted. This also widens the secure zone, to include the access port.

## Address Count Limits

The address count limit parameter specifies limits for the number of IP and MAC addresses that can be entered in a binding table. The purpose of these limits is to contain the size of the binding table based on the number of known and expected hosts, thus enabling you to take pre-emptive action against rogue hosts or IPs in the network.

At a policy level there are separate limits for the number of IP addresses per port, the number of IPv4 addresses per MAC, and IPv6 addresses per MAC. You can configure or change only the number of IP addresses per port.

### IP per Port

The IP per port option is the total number of IP addresses allowed for a port. The address can be IPv4 or IPv6. When the limit is reached, no further IP addresses (i.e., entries) are added to the binding table.

To configure this parameter in a policy, enter the **limit address-count** *ip-per-port* keyword in device-tracking configuration mode. If you configure a limit that is lower than the currently configured one, then the new (lower) limit is applicable only to new entries. An existing entry remains in the binding table and goes through its binding entry lifecycle.

### IPv4 per MAC and IPv6 per MAC

This refers to the number of IPv4 addresses that can be mapped to one MAC address and the number of IPv6 addresses that can be mapped to one MAC address. When the limit is reached, no further entries can be added to the binding table, and traffic from new hosts will be dropped

**Note** The IPv4 per MAC limit and the IPv6 per MAC limit that is effective on an interface or VLAN is as defined in the policy that is applied. If the policy does not specify a limit, this means that a limit does not exist. You cannot change or configure a limit for IPv4 per MAC or IPv6 per MAC for any kind of policy (programmatic, or custom policy, or default policy).

Enter the **show device-tracking policy** *policy name* to check if a limit exisits.

The following is sample output of a policy where an IPv4 per MAC and an IPv6 per MAC limit exists:

```
Device# show device-tracking policy LISP-DT-GUARD-VLAN
Policy LISP-DT-GUARD-VLAN configuration:
  security-level guard (*)
  <output truncated>

  limit address-count for IPv4 per mac 4 (*)
  limit address-count for IPv6 per mac 12 (*)
  tracking enable

<output truncated>
```

### Address Count Limit and SISF Action on the Binding Table

The address count limit parameter affects actions that are performed on the binding table: If a limit is reached, binding entries are not added to the binding table. (It does not influence packet action).

### Address Count Limit and Interactions with Other SISF Settings

- The limits do not have a hierarchy, but the threshold that is set for each limit affects the others.

  For example, if the IP per port limit is 100, and the IPv4 per MAC limit is one, the limit is reached with a single host's IPv4-MAC binding entry. No further IP entries, which are bound to the same MAC are allowed in the table even though the port has a provision for 99 more IP addresses. Similarly, if the IP per port limit is one, and the IPv4 per MAC limit is 100. The limit is reached with a single host's

IPv4-MAC binding entry. No further IP entries are allowed in the table even though the MAC has a provision for 99 more IP addresses for *that* MAC.

- Global and policy-level limits

  The limits configured with the **device-tracking binding max-entries** command are at the global level, the limits configured with the **limit address-count** command in the device-tracking configuration mode are for a policy, which is at the interface or VLAN level.

  If a policy-level value *and* a globally configured value exists, the creation of binding entries is stopped when *a* limit is reached - this limit can be any one of the global values or the policy-level value.

  If only globally configured values exist, the creation of binding entries is stopped when *a* limit is reached.

  If only a policy-level value exists, the creation of binding entries is stopped when the policy-level limit is reached.

## Tracking

The tracking parameter involves tracking of hosts in the network. In section Polling a Host and Updating the Binding Table Entry, on page 477 above, this is referred to as "polling". It also describes polling behaviour in detail.

To configure polling parameters at the global level, enter the **device-tracking tracking** command in global configuration mode. After you configure this command you still have the flexibility to turn polling on or off, for individual interfaces and VLANs. For this you must enable or disable polling in the policy.

To enable polling in a policy, enter the **tracking enable** keywords in the device-tracking configuration mode. By default, polling is disabled in a policy.

# Guidelines for Policy Creation

- If multiple policies are available on a given target, a system-internal policy priority determines which policy takes precedence.

  A manually created policy has the highest priority. When you want to override the settings of a programmatically created policy, you can create a custom policy, so it has higher priority.

- The parameters of a programmatically created policy cannot be changed. You can configure certain attributes of a custom policy.

# Guidelines for Applying a Policy

- Multiple policies can be attached to the same VLAN.
- If a programmatic policy is attached to a VLAN and you want to change policy settings, create a custom device-tracking policy and attach it to the VLAN.
- When multiple policies with different priorities are attached to the same VLAN, the settings of the policy with the highest priority are effective. The exceptions here are the limit address-count for IPv4 per mac and limit address-count for IPv6 per mac settings - the settings of the policy with the lowest priorty are effective.
- When a device-tracking policy is attached to an interface under a VLAN, the policy settings on the interface take precedence over those on its VLAN; exceptions here are the values for limit address-count

for IPv4 per mac and limit address-count for IPv6 per mac, which are aggregated from the policy on both the interface and VLAN.

- A policy cannot be removed unless the device tracking client feature configuration is removed.

# How to Configure SISF

SISF or SISF-based device-tracking, is disabled by default. You enable it by defining a device-tracking policy and attaching the policy to a specific target. The target could be an interface or a VLAN. There are multiple ways to define a policy and no single method is a preferred or recommended one - use the option that suits your requirements.

| Method of Enabling SISF | Applicable Configuration Tasks | Result |
|---|---|---|
| **Option 1**: Manually, by using interface configuration commands to create and apply the **default** policy to a target. | Applying the Default Device Tracking Policy to a Target, on page 495 | Automatically applies the **default** device tracking policy to the specified target.<br>The **default** policy is a built-in policy with default settings; you cannot change any of the attributes of the default policy. See **Option 2** if you want to configure device tracking policy attributes. |
| **Option 2**: Manually, by using global configuration commands commands to create a custom policy and applying the custom policy to a target. | **1.** Creating a Custom Device Tracking Policy with Custom Settings, on page 496<br>**2.** Attach the custom policy to an interface or VLAN:<br>Attaching a Device Tracking Policy to an Interface, on page 500<br>OR<br>Attaching a Device Tracking Policy to a VLAN, on page 501 | Creates a custom policy with the name and policy parameters you configure, and attaches the policy to the specified target. |
| **Option 3**: Programmatically, by configuring the snooping command. | Enter the **ip dhcp snooping vlan***vlan* command in global configuration mode.<br>Example: Programatically Enabling SISF by Configuring DHCP Snooping, on page 504 | When you configure the command, the system automatically creates policy `DT-PROGRAMMATIC`.<br>Use this method if you want to enable SISF-based device tracking for these clients: IEEE 802.1X, Web authentication, Cisco TrustSec, IP Source Guard, and SANET. |

| Method of Enabling SISF | Applicable Configuration Tasks | Result |
|---|---|---|
| **Option 4**: Programmatically, by configuring Locator ID Separation Protocol (LISP). | Example: Programatically enabling SISF by Configuring LISP (LISP-DT-GUARD-VLAN), on page 506<br>Example: Programatically Enabling SISF by Configuring LISP (LISP-DT-GLEAN-VLAN), on page 505<br>Example: Programatically Enabling SISF by Configuring LISP (LISP-DT-GUARD-VLAN-MULTI-IP), on page 507 | When you configure LISP, the system automatically creates policy `LISP-DT-GUARD-VLAN`, `LISP-DT-GLEAN-VLAN`, or other variants, depending on the software release. |
| **Option 5**: Programmatically, by configuring EVPN VLAN. | Example: Programatically Enabling SISF by Configuring EVPN on VLAN, on page 505 | When you configure EVPN on VLAN, the system automatically creates policy `evpn-sisf-policy`, `evpn-device-track`, or other variants, depending on the software release. |
| **Option 6**: By using an interface template | Using an Interface Template to Enable SISF, on page 502 | By adding the policy to an interface template, you can apply the same policy to multiple targets, without having to create it separately for each target. |
| **Option 7**: Migrating from legacy IPDT and IPv6 Snooping. | Migrating from Legacy IPDT and IPv6 Snooping to SISF-Based Device Tracking, on page 503 | Convert legacy IPDT and IPv6 Snooping configuration to the SISF-based device-tracking commands. |

# Applying the Default Device Tracking Policy to a Target

Beginning in privileged EXEC mode, follow these steps to apply the default device tracking policy to an interface or VLAN:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | Specify an interface or a VLAN<br>• **interface** *interface*<br>• **vlan configuration** *vlan_list*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/1/4`<br>OR<br>`Device(config)# vlan configuration 333` | **interface** *type number*—Specifies the interface and enters interface configuration mode. The device tracking policy will be attached to the specified interface.<br>**vlan configuration** *vlan_list*—Specifies the VLANs and enters VLAN feature configuration mode. The device tracking policy will be attached to the specified VLAN. |
| **Step 4** | **device-tracking**<br>**Example:**<br>`Device(config-if)# device-tracking`<br>OR<br>`Device(config-vlan-config)# device-tracking` | Enables SISF-based device tracking and attaches the default policy it to the interface or VLAN.<br>The default policy is a built-in policy with default settings; none of the attributes of the default policy can be changed. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end`<br>OR<br>`Device(config-vlan-config)# end` | Exits interface configuration mode and returns to privileged EXEC mode.<br>Exits VLAN feature configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show device-tracking policy** *policy-name*<br>**Example:**<br>`Device# show device-tracking policy default` | Displays device-tracking policy configuration, and all the targets it is applied to. |

# Creating a Custom Device Tracking Policy with Custom Settings

Beginning in privileged EXEC mode, follow these steps to create and configure a device tracking policy:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | [**no**] **device-tracking policy** *policy-name*<br>**Example:**<br>`Device(config)# device-tracking policy example_policy` | Creates the policy and enters device-tracking configuration mode. |
| **Step 4** | [**data-glean** \| **default** \| **destination-glean** \| **device-role** \| **distribution-switch** \| **exit** \| **limit** \| **no** \| **prefix-glean** \| **protocol** \| **security-level** \| **tracking** \| **trusted-port** \| **vpc**]<br>**Example:**<br>`Device(config-device-tracking)# destination-glean log-only` | Enter the question mark (?) at the system prompt to obtain a list of available options in this mode. You can configure the following for both IPv4 and IPv6:<br>• (Optional) **data-glean**—Enables learning of addresses from a data packet snooped from a source inside the network and populates the binding table with the data traffic source address. Enter one of these options:<br>  • log-only—Generates a syslog message upon data packet notification<br>  • recovery—Uses a protocol to enable binding table recovery. Enter **NDP** or **DHCP**.<br>• (Optional) **default**—Sets the policy attribute to its default value. You can set these policy attributes to their default values: **data-glean**, **destination-glean**, **device-role**, **limit**, **prefix-glean**, **protocol**, **security-level**, **tracking**, **trusted-port**.<br>• (Optional) **destination-glean**—Populates the binding table by gleaning data traffic destination address. Enter one of these options:<br>  • **log-only**—Generates a syslog message upon data packet notification<br>  • **recovery**—Uses a protocol to enable binding table recovery. Enter **DHCP**.<br>• (Optional) **device-role**—Sets the role of the device attached to the port. It can be a node or a switch. Enter one of these options:<br>  • node—Configures the attached device as a node. This is the default option.<br>  • switch—Configures the attached device as a switch. |

| | Command or Action | Purpose |
|---|---|---|
| | | • (Optional) **distribution-switch**—Although visible on the CLI, this option is not supported. Any configuration settings you make will not take effect.<br>• **exit**—Exits the device-tracking policy configuration mode.<br>• **limit** address-count—Specifies an address count limit per port. The range is 1 to 32000.<br>• **no**—Negates the command or sets it to defaults.<br>• (Optional) **prefix-glean**—Enables learning of prefixes from either IPv6 Router Advertisements or from DHCP-PD. You have the following option:<br>  • (Optional) **only**—Gleans only prefixes and not host addresses.<br>• (Optional) **protocol**—Sets the protocol to glean; by default, all are gleaned. Enter one of these options:<br>  • **arp** [**prefix-list** *name*]—Gleans addresses in ARP packets. Optionally, enter the name of prefix-list that is to be matched.<br>  • **dhcp4** [**prefix-list** *name*]—Glean addresses in DHCPv4 packets. Optionally, enter the name of prefix-list that is to be matched.<br>  • **dhcp6** [**prefix-list** *name*]—Glean addresses in DHCPv6 packets. Optionally, enter the name of prefix-list that is to be matched.<br>  • **ndp** [**prefix-list** *name*]—Glean addresses in NDP packets. Optionally, enter the name of prefix-list that is to be matched.<br>  • **udp** [**prefix-list** *name*]—Although visible on the CLI, this option is not supported. Any configuration settings you make will not take effect. |

| | Command or Action | Purpose |
|---|---|---|
| | | • (Optional) **security-level**—Specifies the level of security enforced by the feature. Enter one of these options:<br>• **glean**—Gleans addresses passively.<br>• **guard**—Inspects and drops un-authorized messages. This is the default.<br>• **inspect**—Gleans and validates messages.<br><br>• (Optional) **tracking**—Specfies a tracking option. Enter one of these options:<br>• **disable** [**stale-lifetime** [*1-86400-seconds* \| **infinite**] ] —Turns of device-tracking.<br>Optionally, you can enter the duration for which the entry is kept inactive before deletion, or keep it permanently inactive.<br>• **enable** [**reachable-lifetime** [*1-86400-seconds* \| **infinite**] ] —Turns on device-tracking.<br>Optionally, you can enter the duration for which the entry is kept reachable, or keep it permanently reachable.<br><br>• (Optional) **trusted-port**—Sets up a trusted port. Disables the guard on applicable targets. Bindings learned through a trusted port have preference over bindings learned through any other port. A trusted port is given preference in case of a collision while making an entry in the table.<br>• (Optional) **vpc**—Although visible on the CLI, this option is not supported. Any configuration settings you make will not take effect. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-device-tracking)# end` | Exits device-tracking configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show device-tracking policy** *policy-name*<br>**Example:** | Displays the device-tracking policy configuration. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# show device-tracking policy example_policy` | |

### What to do next

Attach the policy to an interface or VLAN.

# Attaching a Device Tracking Policy to an Interface

Beginning in privileged EXEC mode, follow these steps to attach a device tracking policy to an interface:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface*<br>**Example:**<br>`Device(config-if)# interface gigabitethernet 1/1/4` | Specifies an interface and enters interface configuration mode. |
| **Step 4** | **device-tracking attach-policy** *policy name*<br>**Example:**<br>`Device(config-if)# device-tracking attach-policy example_policy` | Attaches the device tracking policy to the interface. Device tracking is also supported on EtherChannels.<br>**Note**<br>SISF based device-tracking policies can be disabled only if they are custom policies. Programmatically created policies can be removed only if the corresponding device-tracking client feature configuration is removed. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show device-tracking policies** [**interface** *interface*]<br>**Example:** | Displays policies that match the specified interface type and number. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# show device-tracking policies interface gigabitethernet 1/1/4` | |

# Attaching a Device Tracking Policy to a VLAN

Beginning in privileged EXEC mode, follow these steps to attach a device-tracking policy to VLANs across multiple interfaces:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **vlan configuration** *vlan_list*<br>**Example:**<br>`Device(config)# vlan configuration 333` | Specifies the VLANs to which the device tracking policy will be attached; enters the VLAN interface configuration mode. |
| **Step 4** | **device-tracking attach-policy** *policy_name*<br>**Example:**<br>`Device(config-vlan-config)# device-tracking attach-policy example_policy` | Attaches the device tracking policy to the specified VLANs across all switch interfaces.<br>**Note**<br>SISF based device-tracking policies can be disabled only if they are custom policies. Programmatically created policies can be removed only if the corresponding device-tracking client feature configuration is removed. |
| **Step 5** | **do show device-tracking policies vlan** *vlan-ID*<br>**Example:**<br>`Device(config-vlan-config)# do show device-tracking policies vlan 333` | Verifies that the policy is attached to the specified VLAN, without exiting the VLAN interface configuration mode. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-vlan-config)# end` | Exits VLAN feature configuration mode and returns to privileged EXEC mode. |

# Using an Interface Template to Enable SISF

An interface template is a container of configurations or policies. It provides a mechanism to configure multiple commands at the same time and associate it with a target, such as an interface. Starting with Cisco IOS XE Amsterdam 17.3.1, you can add the **device-tracking policy***policy_name* global configuration command to a template and apply it to multiple targets.

You can also apply the template through 802.1x authentication. During the 802.1x authentication process, you can dynamically assign different templates (and therefore different policies) to different interfaces.

**Note** You can apply only one interface template to one port.

### Before you begin

You have already created a custom policy. See Creating a Custom Device Tracking Policy with Custom Settings, on page 496.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode. Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **template interface***template_name*<br>**Example:**<br>`Device(config)# template interface template_w_sisf` | Creates a template with the name you specify and enters template configuration mode. In the accompanying example, a template called "template_w_sisf" is created. |
| **Step 4** | **device-tracking attach-policy***policy_name*<br>**Example:**<br>`Device (config-template)# device-tracking attach-policy sisf_policy_for_template` | Attaches a policy to the template. SISF-based device-tracking is enabled and the policy is applied wherever the template is applied. |
| **Step 5** | **exit**<br>**Example:**<br>`Device (config-template)# exit` | Exits the template configuration mode and enters the global configuration mode. |
| **Step 6** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/1/4` | Specifies an interface and enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **source template***template_name*<br>**Example:**<br>`Device(config-if)# source template template_w_sisf` | Configures a static binding for an interface template. In the accompaying example, "template_w_sisf" is statically applied to an interface. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits the interface configuration mode and enters the privileged EXEC mode. |
| **Step 9** | **show running-config interface** *type number*<br>**Example:**<br>`Device# show running-config interface gigabitethernet 1/1/4`<br><br>`Building configuration...`<br>`<output truncated>`<br><br>`Current configuration : 71 bytes`<br>`!`<br>`interface GigabitEthernet1/1/14`<br>`source template template_w_sisf`<br>`end`<br><br>`<output truncated>` | Displays the contents of the running configuration. |

# Migrating from Legacy IPDT and IPv6 Snooping to SISF-Based Device Tracking

Based on the legacy configuration that exists on your device, the **device-tracking upgrade-cli** command upgrades your CLI differently. Consider the following configuration scenarios and the corresponding migration results before you migrate your existing configuration.

**Note** You cannot configure a mix of the old IPDT and IPv6 snooping CLI with the SISF-based device tracking CLI.

### Only IPDT Configuration Exists

If your device has only IPDT configuration, running the **device-tracking upgrade-cli** command converts the configuration to use the new SISF policy that is created and attached to the interface. You can then update this SISF policy.

If you continue to use the legacy commands, this restricts you to operate in a legacy mode where only the legacy IPDT and IPv6 snooping commands are available on the device.

### Only IPv6 Snooping Configuration Exists

On a device with existing IPv6 snooping configuration, the old IPv6 Snooping commands are available for further configuration. The following options are available:

- (Recommended) Use the **device-tracking upgrade-cli** command to convert all your legacy configuration to the new SISF-based device tracking commands. After conversion, only the new device tracking commands will work on your device.
- Use the legacy IPv6 Snooping commands for your future configuration and do not run the **device-tracking upgrade-cli** command. With this option, only the legacy IPv6 Snooping commands are available on your device, and you cannot use the new SISF-based device tracking CLI commands.

#### Both IPDT and IPv6 Snooping Configuration Exist

On a device that has both legacy IPDT configuration and IPv6 snooping configuration, you can convert legacy commands to the SISF-based device tracking CLI commands. However, note that only one snooping policy can be attached to an interface, and the IPv6 snooping policy parameters override the IPDT settings.

**Note** If you do not migrate to the new SISF-based commands and continue to use the legacy IPv6 snooping or IPDT commands, your IPv4 device tracking configuration information may be displayed in the IPv6 snooping commands, as the SISF-based device tracking feature handles both IPv4 and IPv6 configuration. To avoid this, we recommend that you convert your legacy configuration to SISF-based device tracking commands.

#### No IPDT or IPv6 Snooping Configuration Exists

If your device has no legacy IP Device Tracking or IPv6 Snooping configurations, you can use only the new SISF-based device tracking commands for all your future configuration. The legacy IPDT commands and IPv6 snooping commands are not available.

# Configuration Examples for SISF

## Example: Programatically Enabling SISF by Configuring DHCP Snooping

The following example shows how to configure the **ip dhcp snooping vlan** *vlan* command in global configuration mode to enable SISF-based device-tracking. When you enable SISF this way, the system creates the `DT-PROGRMMATIC` policy.

Enter the **show device-tracking policy** *policy_name* command in privileged EXEC mode, to display the settings for a `DT-PROGRMMATIC` policy.

**Tip** This is only sample output displaying the settings of a programmatic policy and may change from one release to another. Always use the **show** command, to see the settings of a policy as applicable to the software version running on your device.

```
Device> enable
Device# configure terminal
Device(config)# ip dhcp snooping vlan 10
Device(config)# end

Device# show device-tracking policy DT-PROGRAMMATIC
```

```
Policy DT-PROGRAMMATIC configuration:
  security-level glean (*)
  device-role node
  gleaning from Neighbor Discovery
  gleaning from DHCP
  gleaning from ARP
  gleaning from DHCP4
  NOT gleaning from protocol unkn
  limit address-count for IPv4 per mac 1 (*)
  tracking enable
Policy DT-PROGRAMMATIC is applied on the following targets:
Target      Type    Policy               Feature            Target range
vlan 10     VLAN    DT-PROGRAMMATIC      Device-tracking    vlan all


  note:
  Binding entry Down timer: 24 hours (*)
  Binding entry Stale timer:   24 hours (*)
```

# Example: Programatically Enabling SISF by Configuring EVPN on VLAN

When you configure EVPN, the system automatically creates programmatic policy `evpn-sisf-policy`. Enter the **show device-tracking policy** *policy_name* command in privileged EXEC mode, to display policy settings.

**Tip** This is only sample output displaying the settings of a programmatic policy and may change from one release to another. Always use the **show** command, to see the settings of a policy as applicable to the software version running on your device.

```
Device# show device-tracking policy evpn-sisf-policy

Policy evpn-sisf-policy configuration:
  security-level glean (*)
  device-role node
  gleaning from Neighbor Discovery
  gleaning from DHCP
  gleaning from ARP
  gleaning from DHCP4
  NOT gleaning from protocol unkn
  tracking enable
Policy evpn-sisf-policy is applied on the following targets:
Target      Type    Policy                   Feature            Target range
vlan 10     VLAN    evpn-sisf-policy         Device-tracking    vlan all
  note:
  Binding entry Down timer: 24 hours (*)
  Binding entry Stale timer:   24 hours (*)
```

# Example: Programatically Enabling SISF by Configuring LISP (LISP-DT-GLEAN-VLAN)

The following is sample output of programmatic policy `LISP-DT-GLEAN-VLAN`.

**Note** The system creates `LISP-DT-GUARD-VLAN`, or `LISP-DT-GLEAN-VLAN`, or `LISP-DT-GUARD-VLAN-MULTI-IP` depending on *how* LISP is configured. You cannot change this, but if required you can create a custom policy with custom settings and attach it to the required target.

To display policy settings, enter the **show device-tracking policy** *policy_name* command in privileged EXEC mode.

**Tip** This is only sample output displaying the settings of a programmatic policy and may change from one release to another. Always use the **show** command, to see the settings of a policy as applicable to the software version running on your device.

```
Device# show device-tracking policy LISP-DT-GLEAN-VLAN

Policy LISP-DT-GLEAN-VLAN configuration:
  security-level glean (*)
  device-role node
  gleaning from Neighbor Discovery
  gleaning from DHCP
  gleaning from ARP
  gleaning from DHCP4
  NOT gleaning from protocol unkn
  limit address-count for IPv4 per mac 4 (*)
  limit address-count for IPv6 per mac 12 (*)
  tracking enable
Policy LISP-DT-GUARD-VLAN is applied on the following targets:
Target     Type   Policy                 Feature            Target range
vlan 10    VLAN   LISP-DT-GLEAN-VLAN     Device-tracking    vlan all

  note:
  Binding entry Down timer: 10 minutes (*)
  Binding entry Stale timer:   30 minutes (*)
```

# Example: Programatically enabling SISF by Configuring LISP (LISP-DT-GUARD-VLAN)

The following is sample output of programmatic policy `LISP-DT-GUARD-VLAN`.

**Note** The system creates `LISP-DT-GUARD-VLAN`, or `LISP-DT-GLEAN-VLAN`, or `LISP-DT-GUARD-VLAN-MULTI-IP` depending on *how* LISP is configured. You cannot change this, but if required you can create a custom policy with custom settings and attach it to the required target.

To display policy settings, enter the **show device-tracking policy** *policy_name* command in privileged EXEC mode.

**Tip** This is only sample output displaying the settings of a programmatic policy and may change from one release to another. Always use the show command, to see the settings of a policy as applicable to the software version running on your device.

```
Device# show device-tracking policy LISP-DT-GUARD-VLAN

Policy LISP-DT-GUARD-VLAN configuration:
  security-level guard (*)
  device-role node
  gleaning from Neighbor Discovery
  gleaning from DHCP
  gleaning from ARP
  gleaning from DHCP4
  NOT gleaning from protocol unkn
  limit address-count for IPv4 per mac 4 (*)
  limit address-count for IPv6 per mac 12 (*)
  tracking enable
Policy LISP-DT-GUARD-VLAN is applied on the following targets:
Target      Type    Policy                    Feature            Target range
vlan 10     VLAN    LISP-DT-GUARD-VLAN        Device-tracking    vlan all
  note:
  Binding entry Down timer: 10 minutes (*)
  Binding entry Stale timer:   30 minutes (*)
```

# Example: Programatically Enabling SISF by Configuring LISP (LISP-DT-GUARD-VLAN-MULTI-IP)

The following is sample output of programmatic policy `LISP-DT-GUARD-VLAN-MULTI-IP`.

**Note** The system creates `LISP-DT-GUARD-VLAN`, or `LISP-DT-GLEAN-VLAN`, or `LISP-DT-GUARD-VLAN-MULTI-IP` depending on *how* LISP is configured. You cannot change this, but if required you can create a custom policy with custom settings and attach it to the required target.

To display policy settings, enter the **show device-tracking policy** *policy_name* command in privileged EXEC mode.

**Tip** This is only sample output displaying the settings of a programmatic policy and may change from one release to another. Always use the **show** command, to see the settings of a policy as applicable to the software version running on your device.

```
Device# show device-tracking policy LISP-DT-GUARD-VLAN-MULTI-IP

Device-tracking policy LISP-DT-GUARD-VLAN-MULTI-IP configuration:
  security-level guard
  device-role node
  gleaning from Neighbor Discovery
  gleaning from DHCP6
  gleaning from ARP
  gleaning from DHCP4
  NOT gleaning from protocol unkn
  limit address-count for IPv4 per mac 21 preference high
  limit address-count for IPv6 per mac 84
  origin fabric
  tracking enable reachable-lifetime 240
```

# Example: Mitigating the IPv4 Duplicate Address Problem

For an IPv4 device-tracking entry, its reachability is verified by sending an SISF probe to its end-node, which is an ARP request message. Selection of the source IP address for this ARP probe follows these rules:

- If an SVI is configured on the VLAN, the IPv4 address of the SVI is selected. Please ensure that the SVI IP address is unique in the subnet.
- If SVI does not exist and the **device-tracking tracking auto-source** [**fallback** *host-ip mask*] [**override**] command is configured, source IP is selected according to the table below.
- Otherwise, an address with all zeros (0.0.0.0) is selected.

This example shows how you can tackle the `Duplicate IP Address 0.0.0.0 error message` problem encountered by clients that run Microsoft Windows:

Configure the **device-tracking tracking auto-source** command in global configuration mode. This command determines the source IP and MAC address used in the Address Resolution Packet (ARP) request sent by the switch to probe a client, in order to maintain its entry in the device-tracking table. The purpose, is to avoid using 0.0.0.0 as source IP address.

**Note** Configure the **device-tracking tracking auto-source** command when a switch virtual interface (SVI) is not configured. You do not have to configure it when a SVI is configured with an IPv4 address on the VLAN.

| Command | Action<br>(In order to select source IP and MAC address for device tracking ARP probe) | Notes |
|---|---|---|
| **device-tracking tracking auto-source** | • Set source to VLAN SVI if present.<br>• Look for IP and MAC binding in device-tracking table from same subnet.<br>• Use 0.0.0.0 | We recommend that you disable device-tracking on all trunk ports to avoid MAC flapping. |
| **device-tracking tracking auto-source override** | • Set source to VLAN SVI if present<br>• Use 0.0.0.0 | Not recommended when there is no SVI. |

| Command | Action<br>(In order to select source IP and MAC address for device tracking ARP probe) | Notes |
|---|---|---|
| **ip device tracking probe auto-source fallback 0.0.0.X 255.255.255.0** | • Set source to VLAN SVI if present.<br>• Look for IP and MAC binding in device-tracking table from same subnet.<br>• Compute source IP from client IP using host bit and mask provided. Source MAC is taken from the MAC address of the switchport facing the client*. | We recommend that you disable device-tracking on all trunk ports to avoid MAC flapping.<br>The computed IPv4 address must not be assigned to any client or network device. |
| **device-tracking tracking auto-source fallback 0.0.0.X 255.255.255.0 override** | • Set source to VLAN SVI if present.<br>Compute source IP from client IP using host bit and mask provided*. Source MAC is taken from the MAC address of the switchport facing the client*. | |

* Depending on the client IP address, an IPv4 address has to be reserved for the source IP.

A reserved source IPv4 address = (host-ip and mask) | client-ip

- Client IP = 192.0.2.25
- Source IP = (192.0.2.25 and 255.255.255.0) | (0.0.0.1) = 192.0.2.1

IP address 192.0.2.1 should not be assigned to any client or network device.

# Example: Disabling IPv6 Device Tracking on a Target

By default, SISF-based device-tracking supports both IPv4 and IPv6. The following configuration examples show how you can disable IPv6 device-tracking where supported.

To disable device-tracking for IPv6, when a *custom* policy is attached to a target (all releases):

```
Device(config)# device-tracking policy example-policy
 Device(config-device-tracking)# no protocol ndp
 Device(config-device-tracking)# no protocol dhcp6
 Device(config-device-tracking)# end
```

To disable device-tracking for IPv6, when a *programmatic* policy is attached to a target (Only Cisco IOS XE Everest 16.6.x and Cisco IOS XE Fuji 16.8.x):

```
Device(config)# device-tracking policy DT-PROGRAMMATIC
Device(config-device-tracking)# no protocol ndp
Device(config-device-tracking)# no protocol dhcp6
Device(config-device-tracking)# end
```

**Note**

- In the Cisco IOS XE Everest 16.5.x release, when a programmatic policy is attached, you cannot disable device-tracking for IPv6.
- In the Cisco IOS XE Everest 16.6.x and Cisco IOS XE Fuji 16.8.x, when a programmatic policy is attached, you can disable device-tracking for IPv6 - as shown in the example above.
- Starting with Cisco IOS XE Fuji 16.9.x, you cannot change the settings of a programmatic policy.

# Example: Enabling IPv6 for SVI on VLAN (To Mitigate the Duplicate Address Problem)

For an IPv6 device-tracking entry, its reachability is verified by sending an SISF probe to its end-node, which is a neighbor solicitation message. Selection of the source IP address for this neighbor solicitation probe follows these rules:

- If an SVI is configured on the VLAN, the link-local IPv6 address of the SVI is selected. Please ensure that the SVI IP address is unique in the subnet.
- Otherwise, an address with all zeros (0:0:0:0:0:0:0:0) is selected.

When IPv6 is enabled in the network and a switched virtual interface (SVI) is configured on a VLAN, we recommend that you add the following to the SVI configuration. This enables the SVI to acquire a link-local address automatically; this address is used as the source IP address of the SISF probe, thus preventing the duplicate IP address issue.

```
Device> enable
Device# configure terminal
Device(config)# interface vlan 10
Device(config-if)# ipv6 enable
Device(config-if)# end
```

# Example: Configuring a Multi-Switch Network to Stop Creating Binding Entries from a Trunk Port

In a multi-switch network, SISF-based device tracking provides the capability to distribute binding table entries between switches running the feature. Binding entries are only created on the switches where the host appears on an access port. No entry is created for a host that appears over a trunk port. This is achieved by configuring a policy with the **trusted-port** and **device-role switch** options, and attaching it to the trunk port.

**Note** Both, the **trusted-port**, and **device-role switch** options, must be configured in the policy.

Further, we recommended that you apply such a policy on a port facing a device, which also has SISF-based device tracking enabled.

```
Device> enable
Device# configure terminal
Device(config)# device-tracking policy example_trusted_policy
Device(config-device-tracking)# device-role switch
Device(config-device-tracking)# trusted-port
Device(config-device-tracking)# exit
Device(config)# interface gigabitethernet 1/0/25
Device(config-if)# device-tracking attach-policy  example_trusted_policy
Device(config-if)# end
```

## Example: Avoiding a Short Device-Tracking Binding Reachable Time

When migrating from an older release, the following configuration may be present:

```
device-tracking binding reachable-lifetime 10
```

Remove this by entering the **no** version of the command.

```
Device> enable
Device# configure terminal
Device(config)# no device-tracking binding reachable-lifetime 10
Device(config)# end
```

# Feature History for SISF

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | SISF-Based Device-Tracking | This feature was introduced.<br>SISF-Based Device-Tracking tracks the presence, location, and movement of end-nodes in the network. The feature snoops traffic received by the switch, extracts device identity (MAC and IP address), and stores them in a binding table. Other features (called device tracking clients) depend on the accuracy of this information to operate properly.<br>Both IPv4 and IPv6 are supported.<br>SISF-based device-tracking is disabled by default. |

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.6.1 | Option to change parameters of DT_PROGRAMMATIC | Starting with this release, you can change certain settings of the programmatically created device tracking policy: DT_PROGRAMMATIC, in the device tracking configuration mode (config-device-tracking)). |
| Cisco IOS XE Fuji 16.8.1a | SISF-Based Device Tracking on the High Performance models in the series | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.9.1 | Policy priority<br>Additional device tracking clients<br>Change for programmatically created policies | Support for policy priority was introduced. Priority is determined by how the policy is created. A manually created policy has the highest priority. This enables you to apply policy settings that are different from policies that are generated programmatically.<br>More device tracking client features were introduced. The programmatic policy created by each device tracking client differs.<br>The option to change the parameters of *any* programmatic policy was deprecated. |
| Cisco IOS XE Amsterdam 17.3.1 | Interface template | The option to enable SISF-Based Device-Tracking by using an interface template was introduced.<br>You can add the **device-tracking policy** *policy_name* global configuration command to a template and apply it to multiple targets. |
| | Throttling Limit for ARP Packets | ARP packets are throttled to mitigate high CPU utilization scenarios. In a five second window, a maximum of 50 ARP broadcast packets per binding entry are processed by SISF. See the *Binding Table Sources* section in this chapter for more information. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 28

# Configuring IEEE 802.1x Port-Based Authentication

This chapter describes how to configure IEEE 802.1x port-based authentication. IEEE 802.1x authentication prevents unauthorized devices (clients) from gaining access to the network. Unless otherwise noted, the term *switch* refers to a standalone switch or a switch stack.


- Restrictions for IEEE 802.1x Port-Based Authentication, on page 513
- Information About IEEE 802.1x Port-Based Authentication, on page 514
- How to Configure IEEE 802.1x Port-Based Authentication, on page 547
- Configuration Examples for IEEE 802.1x Port-Based Authentication, on page 592
- Monitoring IEEE 802.1x Port-Based Authentication Statistics and Status, on page 593
- Feature History for IEEE 802.1x Port-Based Authentication, on page 594


## Restrictions for IEEE 802.1x Port-Based Authentication

- Switchports are always unauthorized when used with private VLANs. Dynamic VLANs pushed from the Authentication, Authorization, and Accounting (AAA) server is not supported on private VLAN ports. The data client session is expected to authorize on the secondary VLAN of the private VLAN dot1x port.
- Only interface-configured private VLAN-based authorization and dynamic VLAN on a normal access VLAN port is supported.
- If the **dot1q tag vlan native** command is configured globally, the dot1x reauthentication will fail on trunk ports.
- Do not configure the same VLAN ID for both voice VLAN and access VLAN at the same time, because it may cause authentication failures.
- If a downloadable ACL contains any type of duplicate entries, the entries are not auto merged. As a result, the 802.1X session authorization fails. Ensure that the downloadable ACL is optimized without any duplicate entries, for example port-based and name-based entries for the same port.
- Port security is not supported with IEEE 802.1x port-based authentication.
- If you overwrite the running configuration of interfaces with a configuration file loaded in flash, some ports may fail to authenticate the endpoints.

- Wake-on-LAN (WoL) feature is not supported on the Cisco Catalyst 9500 High Performance Series Switches.
- Reauthentication using SNMP MIB is not supported for 802.1x and MAC authentication bypass (MAB) authentication.

# Information About IEEE 802.1x Port-Based Authentication

The 802.1x standard defines a client-server-based access control and authentication protocol that prevents unauthorized clients from connecting to a LAN through publicly accessible ports unless they are properly authenticated. The authentication server authenticates each client connected to a switch port before making available any services offered by the switch or the LAN.

**Note** TACACS is not supported with 802.1x authentication.

Until the client is authenticated, 802.1x access control allows only Extensible Authentication Protocol over LAN (EAPOL), Cisco Discovery Protocol, and Spanning Tree Protocol (STP) traffic through the port to which the client is connected. After authentication is successful, normal traffic can pass through the port.

The table shown below lists the maximum number of session each client supports:

| Client session | Maximum sessions supported |
|---|---|
| Maximum dot1x or MAB client sessions | 2000 |
| Maximum web-based authentication sessions | 2000 |
| Maximum dot1x sessions with critical-auth VLAN enabled and server re-initialized | 2000 |
| Maximum MAB sessions with various session features applied | 2000 |
| Maximum dot1x sessions with service templates or session features applied | 2000 |

## Overview of IEEE 802.1x Port-Based Authentication

The 802.1x standard defines a client-server-based access control and authentication protocol that prevents unauthorized clients from connecting to a LAN through publicly accessible ports unless they are properly authenticated. The authentication server authenticates each client connected to a switch port before making available any services offered by the switch or the LAN.

**Note** TACACS is not supported with 802.1x authentication.

Until the client is authenticated, 802.1x access control allows only Extensible Authentication Protocol over LAN (EAPOL), Cisco Discovery Protocol, and Spanning Tree Protocol (STP) traffic through the port to which the client is connected. After authentication is successful, normal traffic can pass through the port.

# Port-Based Authentication Process

To configure IEEE 802.1X port-based authentication, you must enable authentication, authorization, and accounting (AAA) and specify the authentication method list. A method list describes the sequence and authentication method to be queried to authenticate a user.

The AAA process begins with authentication. When 802.1x port-based authentication is enabled and the client supports 802.1x-compliant client software, these events occur:

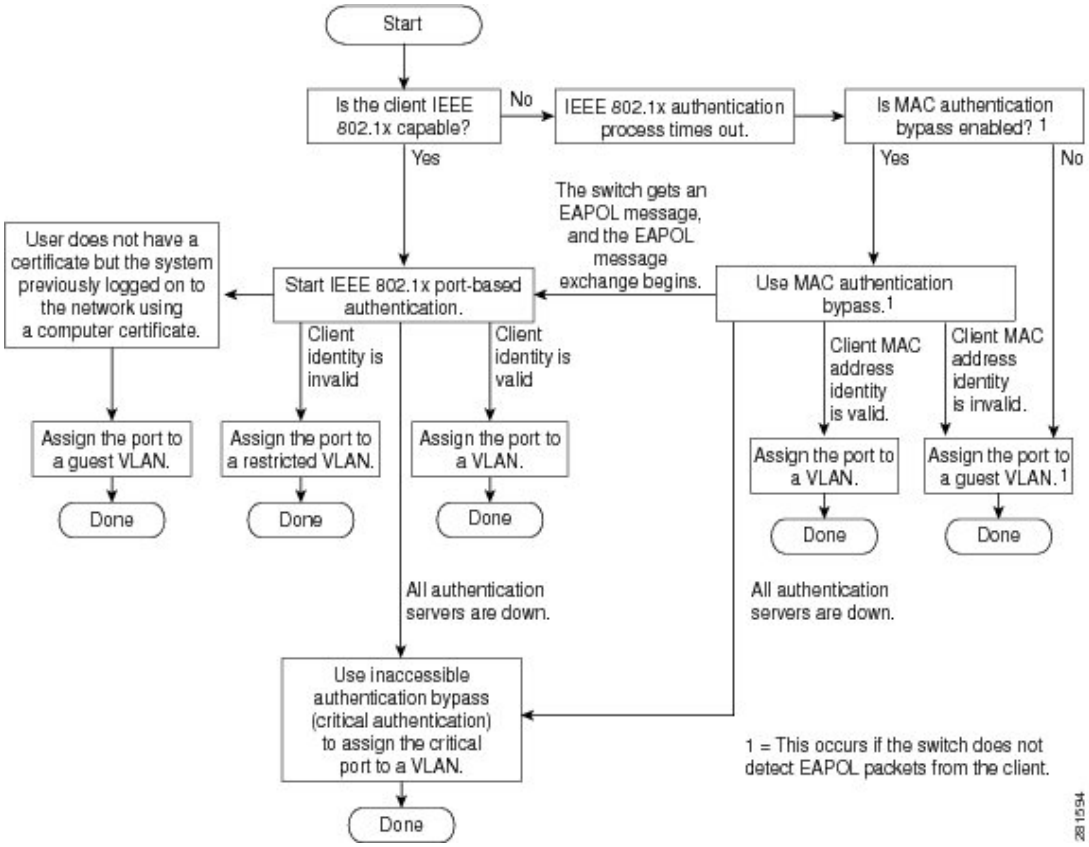
- If the client identity is valid and the 802.1x authentication succeeds, the switch grants the client access to the network.
- If 802.1x authentication times out while waiting for an EAPOL message exchange and MAC authentication bypass is enabled, the switch can use the client MAC address for authorization. If the client MAC address is valid and the authorization succeeds, the switch grants the client access to the network. If the client MAC address is invalid and the authorization fails, the switch assigns the client to a guest VLAN that provides limited services if a guest VLAN is configured.
- If the switch gets an invalid identity from an 802.1x-capable client and a restricted VLAN is specified, the switch can assign the client to a restricted VLAN that provides limited services.
- If the RADIUS authentication server is unavailable (down) and inaccessible authentication bypass is enabled, the switch grants the client access to the network by putting the port in the critical-authentication state in the RADIUS-configured or the user-specified access VLAN.

**Note** Inaccessible authentication bypass is also referred to as critical authentication or the AAA fail policy.

If Multi Domain Authentication (MDA) is enabled on a port, this flow can be used with some exceptions that are applicable to voice authorization.

*Figure 34: Authentication Flowchart*

This figure shows the authentication process.




The switch reauthenticates a client when one of these situations occurs:

- Periodic reauthentication is enabled, and the reauthentication timer expires.

  You can configure the reauthentication timer to use a switch-specific value or to be based on values from the RADIUS server.

  After 802.1x authentication using a RADIUS server is configured, the switch uses timers based on the Session-Timeout RADIUS attribute (Attribute[27]) and the Termination-Action RADIUS attribute (Attribute [29]).

  The Session-Timeout RADIUS attribute (Attribute[27]) specifies the time after which reauthentication occurs. The range is 1 to 1073741823 seconds.

**Note** In releases prior to Cisco IOS XE Bengaluru 17.5.1, the supported timeout range is 1to 65535. seconds. While downgrading from or releases after Cisco IOS XE Bengaluru 17.5.1 set the timeout to supported values to avoid ISSD breakage.

seconds. While downgrading from or releases after Cisco IOS XE Bengaluru 17.5.1 set the timeout to supported values to avoid ISSD breakage.

The Termination-Action RADIUS attribute (Attribute [29]) specifies the action to take during reauthentication. The actions are *Initialize* and *ReAuthenticate*. When the *Initialize* action is set (the attribute value is *DEFAULT*), the 802.1x session ends, and connectivity is lost during reauthentication. When the *ReAuthenticate* action is set (the attribute value is RADIUS-Request), the session is not affected during reauthentication.

- You manually reauthenticate the client by entering the **dot1x re-authenticate interface** *interface-id* privileged EXEC command.

# Port-Based Authentication Initiation and Message Exchange

During 802.1x authentication, the switch or the client can initiate authentication. If you enable authentication on a port by using the **authentication port-control auto** interface configuration command, the switch initiates authentication when the link state changes from down to up or periodically as long as the port remains up and unauthenticated. The switch sends an EAP-request/identity frame to the client to request its identity. Upon receipt of the frame, the client responds with an EAP-response/identity frame.

However, if during bootup, the client does not receive an EAP-request/identity frame from the switch, the client can initiate authentication by sending an EAPOL-start frame, which prompts the switch to request the client's identity.

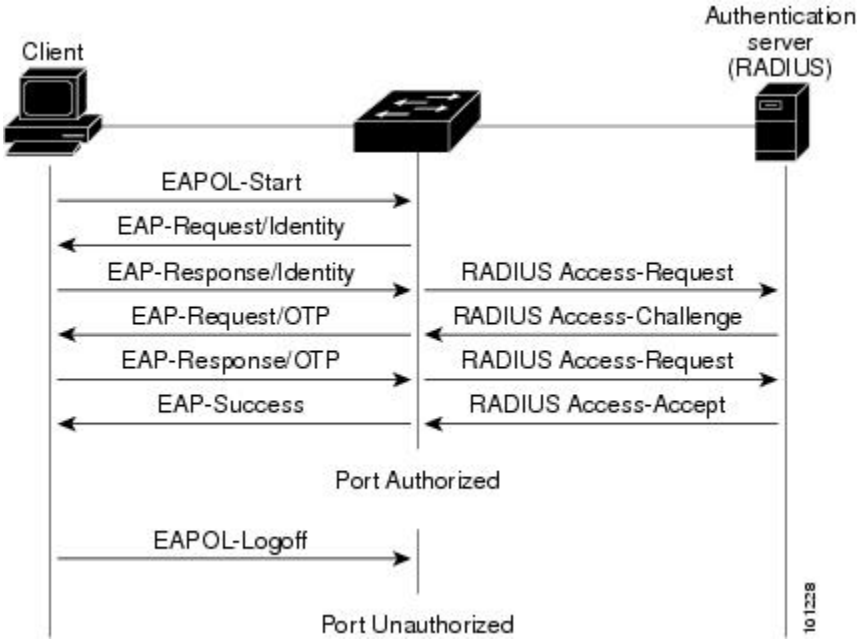
**Note** If 802.1x authentication is not enabled or supported on the network access device, any EAPOL frames from the client are dropped. If the client does not receive an EAP-request/identity frame after three attempts to start authentication, the client sends frames as if the port is in the authorized state. A port in the authorized state effectively means that the client has been successfully authenticated.

When the client supplies its identity, the switch begins its role as the intermediary, passing EAP frames between the client and the authentication server until authentication succeeds or fails. If the authentication succeeds, the switch port becomes authorized. If the authentication fails, authentication can be retried, the port might be assigned to a VLAN that provides limited services, or network access is not granted.

The specific exchange of EAP frames depends on the authentication method being used.

*Figure 35: Message Exchange*

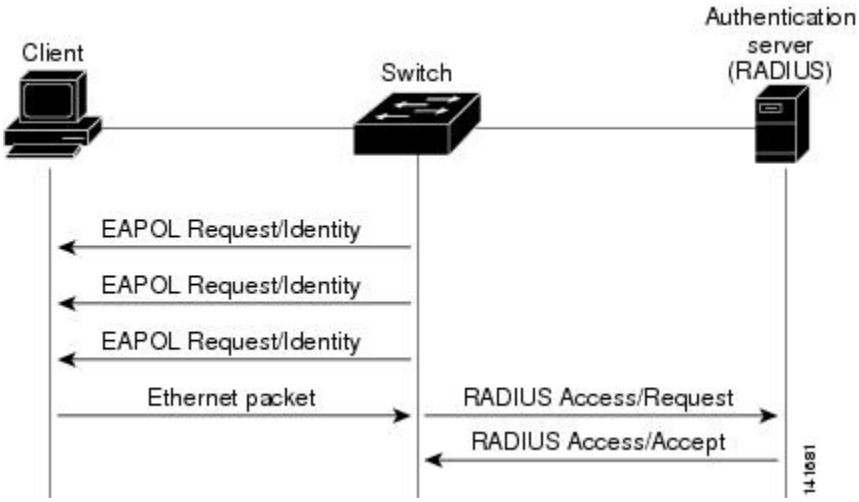
This figure shows a message exchange initiated by the client when the client uses the One-Time-Password (OTP) authentication method with a RADIUS server.




If 802.1x authentication times out while waiting for an EAPOL message exchange and MAC authentication bypass is enabled, the switch can authorize the client when the switch detects an Ethernet packet from the client. The switch uses the MAC address of the client as its identity and includes this information in the RADIUS-access/request frame that is sent to the RADIUS server. After the server sends the switch the RADIUS-access/accept frame (authorization is successful), the port becomes authorized. If authorization fails and a guest VLAN is specified, the switch assigns the port to the guest VLAN. If the switch detects an EAPOL packet while waiting for an Ethernet packet, the switch stops the MAC authentication bypass process and starts 802.1x authentication.

*Figure 36: Message Exchange During MAC Authentication Bypass*

This figure shows the message exchange during MAC authentication bypass.



# Port-Based Authentication Methods

*Table 26: 802.1x Features*

| Authentication method | Mode | | | |
|---|---|---|---|---|
| | **Single host** | **Multiple host** | **MDA** | **Multiple Authentication** |
| 802.1x | VLAN assignment<br>Per-user ACL<br>Filter-ID attribute<br>Downloadable ACL<br>Redirect URL | VLAN assignment | VLAN assignment<br>Per-user ACL<br>Filter-Id attribute<br>Downloadable ACL<br>Redirect URL | VLAN assignment<br>Per-user ACL<br>Filter-Id attribute<br>Downloadable ACL<br>Redirect URL |
| MAC authentication bypass | VLAN assignment<br>Per-user ACL<br>Filter-ID attribute<br>Downloadable ACL<br>Redirect URL | VLAN assignment | VLAN assignment<br>Per-user ACL<br>Filter-Id attribute<br>Downloadable ACL<br>Redirect URL | VLAN assignment<br>Per-user ACL<br>Filter-Id attribute<br>Downloadable ACL<br>Redirect URL |
| Standalone web authentication | Proxy ACL, Filter-Id attribute, downloadable ACL | | | |
| NAC Layer 2 IP validation | Filter-Id attribute<br>Downloadable ACL<br>Redirect URL | Filter-Id attribute<br>Downloadable ACL<br>Redirect URL | Filter-Id attribute<br>Downloadable ACL<br>Redirect URL | Filter-Id attribute<br>Downloadable ACL<br>Redirect URL |
| Web authentication as fallback method | Proxy ACL<br>Filter-Id attribute<br>Downloadable ACL | Proxy ACL<br>Filter-Id attribute<br>Downloadable ACL | Proxy ACL<br>Filter-Id attribute<br>Downloadable ACL | Proxy ACL<br>Filter-Id attribute<br>Downloadable ACL |

[3] Supported in Cisco IOS Release 12.2(50)SE and later.
[4] For clients that do not support 802.1x authentication.

# Per-User ACLs and Filter IDs

**Note** Using role-based ACLs as Filter ID is not recommended.

More than one host can be authenticated on MDA-enabled and multiauth ports. The ACL policy applied for one host does not effect the traffic of another host. If only one host is authenticated on a multi-host port, and

the other hosts gain network access without authentication, the ACL policy for the first host can be applied to the other connected hosts by specifying any in the source address.

# Ports in Authorized and Unauthorized States

During 802.1x authentication, depending on the switch port state, the switch can grant a client access to the network. The port starts in the *unauthorized* state. While in this state, the port that is not configured as a voice VLAN port disallows all ingress and egress traffic except for 802.1x authentication, Cisco Discovery Protocol, and STP packets. When a client is successfully authenticated, the port changes to the *authorized* state, allowing all traffic for the client to flow normally. If the port is configured as a voice VLAN port, the port allows VoIP traffic and 802.1x protocol packets before the client is successfully authenticated.

**Note** Cisco Discovery Protocol bypass is not supported and may cause a port to go into err-disabled state.

If a client that does not support 802.1x authentication connects to an unauthorized 802.1x port, the switch requests the client's identity. In this situation, the client does not respond to the request, the port remains in the unauthorized state, and the client is not granted access to the network.

In contrast, when an 802.1x-enabled client connects to a port that is not running the 802.1x standard, the client initiates the authentication process by sending the EAPOL-start frame. When no response is received, the client sends the request for a fixed number of times. Because no response is received, the client begins sending frames as if the port is in the authorized state.

You control the port authorization state by using the **authentication port-control** interface configuration command and these keywords:

- **force-authorized**: Disables 802.1x authentication and causes the port to change to the authorized state without any authentication exchange required. The port sends and receives normal traffic without 802.1x-based authentication of the client. This is the default setting.
- **force-unauthorized**: Causes the port to remain in the unauthorized state, ignoring all attempts by the client to authenticate. The switch cannot provide authentication services to the client through the port.
- **auto**: Enables 802.1x authentication and causes the port to begin in the unauthorized state, allowing only EAPOL frames to be sent and received through the port. The authentication process begins when the link state of the port changes from down to up or when an EAPOL-start frame is received. The switch requests the identity of the client and begins relaying authentication messages between the client and the authentication server. Each client attempting to access the network is uniquely identified by the switch by using the client MAC address.

If the client is successfully authenticated (receives an Accept frame from the authentication server), the port state changes to authorized, and all frames from the authenticated client are allowed through the port. If the authentication fails, the port remains in the unauthorized state, but authentication can be retried. If the authentication server cannot be reached, the switch can resend the request. If no response is received from the server after the specified number of attempts, authentication fails, and network access is not granted.

When a client logs off, it sends an EAPOL-logoff message, causing the switch port to change to the unauthorized state.

If the link state of a port changes from up to down, or if an EAPOL-logoff frame is received, the port returns to the unauthorized state.

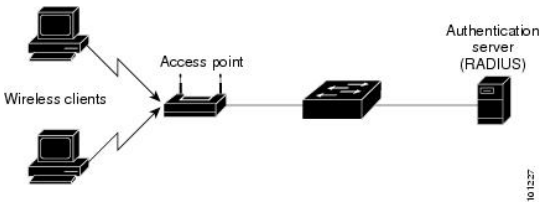
# 802.1x Host Mode

You can configure an 802.1x port for single-host or for multiple-hosts mode. In single-host mode, only one client can be connected to the 802.1x-enabled switch port. The switch detects the client by sending an EAPOL frame when the port link state changes to the up state. If a client leaves or is replaced with another client, the switch changes the port link state to down, and the port returns to the unauthorized state.

In multiple-hosts mode, you can attach multiple hosts to a single 802.1x-enabled port. In this mode, only one of the attached clients must be authorized for all clients to be granted network access. If the port becomes unauthorized (reauthentication fails or an EAPOL-logoff message is received), the switch denies network access to all of the attached clients.

In this topology, the wireless access point is responsible for authenticating the clients attached to it, and it also acts as a client to the switch.

***Figure 37: Multiple Host Mode Example***




**Note** For all host modes, the line protocol stays up before authorization when port-based authentication is configured.

The switch supports multidomain authentication (MDA), which allows both a data device and a voice device, such as an IP Phone, to connect to the same switch port.

# Access Session Limit Profile

An access session limit profile will allow you to limit the number of voice and data hosts connecting to a port. An access session limit profile will have higher priority compared to any host mode configuration. When an access session limit profile is configured the host mode configuration will be ignored.

You can create an access session limit profile by using the **access-session limit profile** command in the global configuration mode. You can configure the profile to limit the number of data and voice sessions allowed per interface. The profile can be configured to allow multiple hosts and to bypass authentication based on CDP packets if CDP bypass is supported.

The access session limit profile needs to be applied at an interface level.

You can also attach the access session limit profile to an interface template.

# MAC Move

When a MAC address is authenticated on one switch port, that address is not allowed on another authentication manager-enabled port of the switch. If the switch detects that same MAC address on another authentication manager-enabled port, the address is not allowed.

There are situations where a MAC address might need to move from one port to another on the same switch. For example, when there is another device (for example a hub or an IP phone) between an authenticated host and a switch port, you might want to disconnect the host from the device and connect it directly to another port on the same switch.

You can globally enable MAC move so the device is reauthenticated on the new port. When a host moves to a second port, the session on the first port is deleted, and the host is reauthenticated on the new port. MAC move is supported on all host modes. (The authenticated host can move to any port on the switch, no matter which host mode is enabled on that port.) When a MAC address moves from one port to another, the switch terminates the authenticated session on the original port and initiates a new authentication sequence on the new port. The MAC move feature applies to both voice and data hosts.

**Note** In open authentication mode, a MAC address is immediately moved from the original port to the new port, with no requirement for authorization on the new port.

MAC move with port security is not supported.

MAC move from an authentication manager-enabled port to an authentication manager-disabled port on a device is enabled by default. For this to work, MAC move must be enabled globally, that is, the **authentication mac-move permit** command must be configured in legacy mode, or, the **no access-session mac-move deny** command must be configured in IBNS 2.0. When MAC move is disabled globally, MAC move is denied. To disable MAC move from an authentication manager-enabled port to an authentication manager-disabled port on a device, use the **authentication mac-move deny-uncontrolled** command.

# MAC Replace

The MAC Replace feature can be configured to address the violation that occurs when a host attempts to connect to a port where another host was previously authenticated.

**Note** This feature does not apply to ports in multi-auth mode, because violations are not triggered in that mode. It does not apply to ports in multiple host mode, because in that mode, only the first host requires authentication.

If you configure the **authentication violation** interface configuration command with the **replace** keyword, the authentication process on a port in multidomain mode is:

- A new MAC address is received on a port with an existing authenticated MAC address.
- The authentication manager replaces the MAC address of the current data host on the port with the new MAC address.
- The authentication manager initiates the authentication process for the new MAC address.
- If the authentication manager determines that the new host is a voice host, the original voice host is removed.

If a port is in open authentication mode, any new MAC address is immediately added to the MAC address table.

# 802.1x Accounting

The 802.1x standard defines how users are authorized and authenticated for network access but does not keep track of network usage. 802.1x accounting is disabled by default. You can enable 802.1x accounting to monitor this activity on 802.1x-enabled ports:

- User successfully authenticates.
- User logs off.
- Link-down occurs.
- Reauthentication successfully occurs.
- Reauthentication fails.

The switch does not log 802.1x accounting information. Instead, it sends this information to the RADIUS server, which must be configured to log accounting messages.

# 802.1x Accounting Attribute-Value Pairs

The information sent to the RADIUS server is represented in the form of Attribute-Value (AV) pairs. These AV pairs provide data for different applications. (For example, a billing application might require information that is in the Acct-Input-Octets or the Acct-Output-Octets attributes of a RADIUS packet.)

AV pairs are automatically sent by a switch that is configured for 802.1x accounting. Three types of RADIUS accounting packets are sent by a switch:

- START: Sent when a new user session starts
- INTERIM: Sent during an existing session for updates
- STOP: Sent when a session terminates

This table lists the AV pairs and when they are sent are sent by the switch.

***Table 27: Accounting AV Pairs***

| Attribute Number | AV Pair Name | START | INTERIM | STOP |
|---|---|---|---|---|
| Attribute[1] | User-Name | Always | Always | Always |
| Attribute[4] | NAS-IP-Address | Always | Always | Always |
| Attribute[5] | NAS-Port | Always | Always | Always |
| Attribute[8] | Framed-IP-Address | Never | Sometimes[5] | Sometimes |
| Attribute[30] | Called-Station-ID | Always | Always | Always |
| Attribute[31] | Calling-Station-ID | Always | Always | Always |
| Attribute[40] | Acct-Status-Type | Always | Always | Always |
| Attribute[41] | Acct-Delay-Time | Always | Always | Always |

| Attribute Number | AV Pair Name | START | INTERIM | STOP |
|---|---|---|---|---|
| Attribute[42] | Acct-Input-Octets | Never | Always | Always |
| Attribute[43] | Acct-Output-Octets | Never | Always | Always |
| Attribute[47] | Acct-Input-Packets | Never | Always | Always |
| Attribute[48] | Acct-Output-Packets | Never | Always | Always |
| Attribute[44] | Acct-Session-ID | Always | Always | Always |
| Attribute[45] | Acct-Authentic | Always | Always | Always |
| Attribute[46] | Acct-Session-Time | Never | Never | Always |
| Attribute[49] | Acct-Terminate-Cause | Never | Never | Always |
| Attribute[61] | NAS-Port-Type | Always | Always | Always |

[5] The Framed-IP-Address AV pair is sent when a valid static IP address is configured or w when a Dynamic Host Control Protocol (DHCP) binding exists for the host in the DHCP snooping bindings table.

# 802.1x Readiness Check

The 802.1x readiness check monitors 802.1x activity on all the switch ports and displays information about the devices connected to the ports that support 802.1x. You can use this feature to determine if the devices connected to the switch ports are 802.1x-capable. You use an alternate authentication such as MAC authentication bypass or web authentication for the devices that do not support 802.1x functionality.

This feature only works if the supplicant on the client supports a query with the NOTIFY EAP notification packet. The client must respond within the 802.1x timeout value.

# Switch-to-RADIUS Server Communication

RADIUS security servers are identified by their hostname or IP address, hostname and specific UDP port numbers, or IP address and specific UDP port numbers. The combination of the IP address and UDP port number creates a unique identifier, which enables RADIUS requests to be sent to multiple UDP ports on a server at the same IP address. If two different host entries on the same RADIUS server are configured for the same service, for example, authentication, the second host entry configured acts as the fail-over backup to the first one. The RADIUS host entries are tried in the order that they were configured.

# IEEE 802.1x Authentication

The following sections provide information about IEEE 802.1x authentication.

## 802.1x Authentication

These are the 802.1x authentication configuration guidelines:

- You must enable SISF-Based device tracking to use 802.1x authentication. By default, SISF-Based device tracking is disabled on a switch.

- When 802.1x authentication is enabled, ports are authenticated before any other Layer 2 or Layer 3 features are enabled.
- If the VLAN to which an 802.1x-enabled port is assigned changes, this change is transparent and does not affect the switch. For example, this change occurs if a port is assigned to a RADIUS server-assigned VLAN and is then assigned to a different VLAN after reauthentication.

  If the VLAN to which an 802.1x port is assigned to shut down, disabled, or removed, the port becomes unauthorized. For example, the port is unauthorized after the access VLAN to which a port is assigned shuts down or is removed.
- The 802.1x protocol is supported on Layer 2 static-access ports, voice VLAN ports, and Layer 3 routed ports, but it is not supported on these port types:
  - Dynamic ports: A port in dynamic mode can negotiate with its neighbor to become a trunk port. If you try to enable 802.1x authentication on a dynamic port, an error message appears, and 802.1x authentication is not enabled. If you try to change the mode of an 802.1x-enabled port to dynamic, an error message appears, and the port mode is not changed.
  - EtherChannel port: Do not configure a port that is an active or a not-yet-active member of an EtherChannel as an 802.1x port. If you try to enable 802.1x authentication on an EtherChannel port, an error message appears, and 802.1x authentication is not enabled.
  - Switched Port Analyzer (SPAN) and Remote SPAN (RSPAN) destination ports: You can enable 802.1x authentication on a port that is a SPAN or RSPAN destination port. However, 802.1x authentication is disabled until the port is removed as a SPAN or RSPAN destination port. You can enable 802.1x authentication on a SPAN or RSPAN source port.
- Before globally enabling 802.1x authentication on a switch by entering the **dot1x system-auth-control** global configuration command, remove the EtherChannel configuration from the interfaces on which 802.1x authentication and EtherChannel are configured.
- Filtering of system messages related to 802.1x authentication is supported.

**Note** We recommend that you configure all the dependent 802.1x CLIs under the same interface or on the same template.

## Port-Based Authentication Manager CLI Commands

The authentication-manager interface-configuration commands control all the authentication methods, such as 802.1x, MAC authentication bypass, and web authentication. The authentication manager commands determine the priority and order of authentication methods applied to a connected host.

The authentication manager commands control generic authentication features, such as host-mode, violation mode, and the authentication timer. Generic authentication commands include the **authentication host-mode**, **authentication violation**, and **authentication timer** interface configuration commands.

802.1x-specific commands begin with the **dot1x** keyword. For example, the **authentication port-control auto** interface configuration command enables authentication on an interface.

To disable dot1x on a switch, remove the configuration globally by using the no dot1x system-auth-control , and also remove it from all configured interfaces.

**Note** If 802.1x authentication is globally disabled, other authentication methods are still enabled on that port, such as web authentication.

The **authentication manager** commands provide the same functionality as earlier 802.1x commands.

When filtering out verbose system messages generated by the authentication manager, the filtered content typically relates to authentication success. You can also filter verbose messages for 802.1x authentication and MAB authentication. There is a separate command for each authentication method:

- The **no authentication logging verbose** global configuration command filters verbose messages from the authentication manager.
- The **no dot1x logging verbose** global configuration command filters 802.1x authentication verbose messages.
- The **no mab logging verbose** global configuration command filters MAC authentication bypass (MAB) verbose messages

## Default 802.1x Authentication Configuration

*Table 28: Default 802.1x Authentication Configuration*

| Feature | Default Setting |
|---|---|
| Switch 802.1x enable state | Disabled. |
| Per-port 802.1x enable state | Disabled (force-authorized).<br>The port sends and receives normal traffic without 802.1x-based authentication of the client. |
| AAA | Disabled. |
| RADIUS server<br>• IP address<br>• UDP authentication port<br>• Default accounting port<br>• Key | • None specified.<br>• 1645.<br>• 1646.<br>• None specified. |
| Host mode | Single-host mode. |
| Control direction | Bidirectional control. |
| Periodic reauthentication | Disabled. |
| Number of seconds between reauthentication attempts | 3600 seconds. |
| Re-authentication number | Twice (number of times that the switch restarts the authentication process before the port changes to the unauthorized state). |

| Feature | Default Setting |
|---|---|
| Quiet period | 60 seconds (number of seconds that the switch remains in the quiet state following a failed authentication exchange with the client). |
| Retransmission time | 30 seconds (number of seconds that the switch should wait for a response to an EAP request/identity frame from the client before resending the request). |
| Maximum retransmission number | 2 times (number of times that the switch will send an EAP-request/identity frame before restarting the authentication process). |
| Client timeout period | 30 seconds (when relaying a request from the authentication server to the client, the amount of time the switch waits for a response before resending the request to the client.) |
| Authentication server timeout period | 30 seconds (when relaying a response from the client to the authentication server, the amount of time the switch waits for a reply before resending the response to the server.)<br>You can change this timeout period by using the dot1x timeout server-timeout interface configuration command. |
| Inactivity timeout | Disabled. |
| Guest VLAN | None specified. |
| Inaccessible authentication bypass | Disabled. |
| Restricted VLAN | None specified. |
| Authenticator (switch) mode | None specified. |
| MAC authentication bypass | Disabled. |
| Voice-aware security | Disabled. |

## Port-Based Authentication and Switch Stacks

If a switch is added to or removed from a switch stack, 802.1x authentication is not affected as long as the IP connectivity between the RADIUS server and the stack remains intact. This statement also applies if the stack's active switch is removed from the switch stack. Note that if the active switch fails, a member switch in the stack becomes the new active switch by using the election process, and the 802.1x authentication process continues as usual.

If IP connectivity to the RADIUS server is interrupted because the switch that was connected to the server is removed or fails, these events occur:

- Ports that are already authenticated and that do not have periodic reauthentication enabled remain in the authenticated state. Communication with the RADIUS server is not required.

- Ports that are already authenticated and that have periodic reauthentication enabled (with the **dot1x re-authentication** global configuration command) fail the authentication process when the reauthentication occurs. Ports return to the unauthenticated state during the reauthentication process. Communication with the RADIUS server is required.

  For an ongoing authentication, the authentication fails immediately because there is no server connectivity.

If the switch that failed comes up and rejoins the switch stack, the authentications might or might not fail depending on the boot-up time and whether the connectivity to the RADIUS server is re-established by the time the authentication is attempted.

To avoid loss of connectivity to the RADIUS server, you should ensure that there is a redundant connection to it. For example, you can have a redundant connection to the active switch and another to a member switch, and if the active switch fails, the switch stack still has connectivity to the RADIUS server.

## 802.1x Authentication with VLAN Assignment

The switch supports 802.1x authentication with VLAN assignment. After successful 802.1x authentication of a port, the RADIUS server sends the VLAN assignment to configure the switch port. The RADIUS server database maintains the username-to-VLAN mappings, assigning the VLAN based on the username of the client connected to the switch port. You can use this feature to limit network access for certain users.

When a voice device is authorized and the RADIUS server returned an authorized VLAN, the voice VLAN on the port is configured to send and receive packets on the assigned voice VLAN. Voice VLAN assignment behaves the same as data VLAN assignment on multidomain authentication (MDA)-enabled ports.

When configured on the switch and the RADIUS server, 802.1x authentication with VLAN assignment has these characteristics:

- If no VLAN is supplied by the RADIUS server or if 802.1x authentication is disabled, the port is configured in its access VLAN after successful authentication. Recall that an access VLAN is a VLAN assigned to an access port. All packets sent from or received on this port belong to this VLAN.
- If 802.1x authentication is enabled but the VLAN information from the RADIUS server is not valid, authorization fails and configured VLAN remains in use. This prevents ports from appearing unexpectedly in an inappropriate VLAN because of a configuration error.

  Configuration errors could include specifying a VLAN for a routed port, a malformed VLAN ID, a nonexistent or internal (routed port) VLAN ID, an RSPAN VLAN, a shut down or suspended VLAN. In the case of a multidomain host port, configuration errors can also be due to an attempted assignment of a data VLAN that matches the configured or assigned voice VLAN ID (or the reverse).
- If 802.1x authentication is enabled and all information from the RADIUS server is valid, the authorized device is placed in the specified VLAN after authentication.
- If the multiple-hosts mode is enabled on an 802.1x port, all hosts are placed in the same VLAN (specified by the RADIUS server) as the first authenticated host.
- Enabling port security does not impact the RADIUS server-assigned VLAN behavior.
- If 802.1x authentication is disabled on the port, it is returned to the configured access VLAN and configured voice VLAN.
- If an 802.1x port is authenticated and put in the RADIUS server-assigned VLAN, any change to the port access VLAN configuration does not take effect. In the case of a multidomain host, the same applies to voice devices when the port is fully authorized with these exceptions:

- If the VLAN configuration change of one device results in matching the other device configured or assigned VLAN, then authorization of all devices on the port is terminated and multidomain host mode is disabled until a valid configuration is restored where data and voice device configured VLANs no longer match.
- If a voice device is authorized and is using a downloaded voice VLAN, the removal of the voice VLAN configuration, or modifying the configuration value to *dot1p* or *untagged* results in voice device unauthorization and the disablement of multi-domain host mode.

When the port is in the force authorized, force unauthorized, unauthorized, or shutdown state, it is put into the configured access VLAN.

To configure VLAN assignment you need to perform these tasks:

- Enable AAA authorization by using the **network** keyword to allow interface configuration from the RADIUS server.
- Enable 802.1x authentication. (The VLAN assignment feature is automatically enabled when you configure 802.1x authentication on an access port).
- Assign vendor-specific tunnel attributes in the RADIUS server. The RADIUS server must return these attributes to the switch:
  - [64] Tunnel-Type = VLAN
  - [65] Tunnel-Medium-Type = 802
  - [81] Tunnel-Private-Group-ID = VLAN name or VLAN ID
  - [83] Tunnel-Preference

  Attribute [64] must contain the value *VLAN* (type 13). Attribute [65] must contain the value *802* (type 6). Attribute [81] specifies the *VLAN name* or *VLAN ID* assigned to the IEEE 802.1x-authenticated user.

## 802.1x Authentication with Per-User ACLs

You can enable per-user access control lists (ACLs) to provide different levels of network access and service to an 802.1x-authenticated user. When the RADIUS server authenticates a user connected to an 802.1x port, it retrieves the ACL attributes based on the user identity and sends them to the switch. The switch applies the attributes to the 802.1x port for the duration of the user session. The switch removes the per-user ACL configuration when the session is over, if authentication fails, or if a link-down condition occurs. The switch does not save RADIUS-specified ACLs in the running configuration. When the port is unauthorized, the switch removes the ACL from the port.

You can configure router ACLs and input port ACLs on the same switch. However, a port ACL takes precedence over a router ACL. If you apply input port ACL to an interface that belongs to a VLAN, the port ACL takes precedence over an input router ACL applied to the VLAN interface. Incoming packets received on the port, to which a port ACL is applied, are filtered by the port ACL. Incoming routed packets received on other ports are filtered by the router ACL. Outgoing routed packets are filtered by the router ACL. To avoid configuration conflicts, you should carefully plan the user profiles stored on the RADIUS server.

RADIUS supports per-user attributes, including vendor-specific attributes. These vendor-specific attributes (VSAs) are in octet-string format and are passed to the switch during the authentication process. The VSAs used for per-user ACLs are inacl#<*n*> for the ingress direction and outacl#<*n*> for the egress direction. MAC

ACLs are supported only in the ingress direction. The switch supports VSAs only in the ingress direction. It does not support port ACLs in the egress direction on Layer 2 ports.

Use only the extended ACL syntax style to define the per-user configuration stored on the RADIUS server. When the definitions are passed from the RADIUS server, they are created by using the extended naming convention. However, if you use the Filter-Id attribute, it can point to a standard ACL.

You can use the Filter-Id attribute to specify an inbound or outbound ACL that is already configured on the switch. The attribute contains the ACL number followed by *.in* for ingress filtering or *.out* for egress filtering. If the RADIUS server does not allow the *.in* or *.out* syntax, the access list is applied to the outbound ACL by default. The user is marked unauthorized if the Filter-Id sent from the RADIUS server is not configured on the device. The Filter-Id attribute is supported only for IP ACLs numbered in the range of 1 to 199 (IP standard ACLs) and 1300 to 2699 (IP extended ACLs).

The maximum size of the per-user ACL is 4000 ASCII characters but is limited by the maximum size of RADIUS-server per-user ACLs.

You must meet the following prerequisites to configure per-user ACLs:

- Enable AAA authentication.
- Enable AAA authorization by using the **network** keyword to allow interface configuration from the RADIUS server.
- Enable 802.1x authentication.
- Configure the user profile and VSAs on the RADIUS server.
- Configure the 802.1x port for single-host mode.

**Note** Per-user ACLs are supported only in single-host mode.

## 802.1x Authentication with Downloadable ACLs and Redirect URLs

You can download ACLs and redirect URLs from a RADIUS server to the switch during 802.1x authentication or MAC authentication bypass of the host. You can also download ACLs during web authentication. A downloadable ACL is also referred to as a *dACL*.

If more than one host is authenticated and the host is in single-host, MDA, or multiple-authentication mode, the switch changes the source address of the ACL to the host IP address.

You can apply the ACLs and redirect URLs to all the devices connected to the 802.1x-enabled port.

If no ACLs are downloaded during 802.1x authentication, the switch applies the static default ACL on the port to the host. On a voice VLAN port configured in multi-auth or MDA mode, the switch applies the ACL only to the phone as part of the authorization policies.

The limit for dACL with stacking is 64 ACEs per dACL per port. The limit without stacking is the number of available TCAM entries which varies based on the other ACL features that are active.

The 802.1x and MAB authentication methods support two authentication modes, *open* and *closed*. If there is no static ACL on a port in *closed* authentication mode:

- When the first host authenticates, the authorization policy is applied without IP address insertion.

- When a second host is detected, the policies for the first host are refreshed, and policies for the first and subsequent sessions are enforced with IP address insertion.

If there is no static ACL on a port in *open* authentication mode:

- Policies are enforced with IP address insertion to prevent security breaches.

To control access for hosts with no authorization policy, you can configure a directive. The supported values for the directive are *open* and *default*. When you configure the *open* directive, all traffic is allowed. The *default* directive subjects traffic to the access provided by the port. You can configure the directive either in the user profile on the AAA server or on the switch. To configure the directive on the AAA server, use the **authz-directive =<open/default>** global command. To configure the directive on the switch, use the **epm access-control open** global configuration command.

**Note** The default value of the directive is *default*.

For a URL redirect ACL:

- Packets that match a permit access control entry (ACE) rule are sent to the CPU for forwarding to the AAA server.
- Packets that match a deny ACE rule are forwarded through the switch.
- Packets that match neither the permit ACE rule or deny ACE rule are processed by the next dACL, and if there is no dACL, the packets hit the implicit-deny ACL and are dropped.

## VLAN ID-Based MAC Authentication

You can use VLAN ID-based MAC authentication if you wish to authenticate hosts based on a static VLAN ID instead of a downloadable VLAN. When you have a static VLAN policy configured on your switch, VLAN information is sent to an IAS (Microsoft) RADIUS server along with the MAC address of each host for authentication. The VLAN ID configured on the connected port is used for MAC authentication. By using VLAN ID-based MAC authentication with an IAS server, you can have a fixed number of VLANs in the network.

The feature also limits the number of VLANs monitored and handled by STP. The network can be managed as a fixed VLAN.

## IEEE 802.1x Authentication with MAC Authentication Bypass

You can configure the switch to authorize clients based on the client MAC address by using the MAC authentication bypass feature. For example, you can enable this feature on IEEE 802.1x ports connected to devices such as printers.

If IEEE 802.1x authentication times out while waiting for an EAPOL response from the client, the switch tries to authorize the client by using MAC authentication bypass.

When the MAC authentication bypass feature is enabled on an IEEE 802.1x port, the switch uses the MAC address as the client identity. The authentication server has a database of client MAC addresses that are allowed network access. After detecting a client on an IEEE 802.1x port, the switch waits for an Ethernet packet from the client. The switch sends the authentication server a RADIUS-access/request frame with a username and password based on the MAC address. If authorization succeeds, the switch grants the client access to the

network. If authorization fails, the switch assigns the port to the guest VLAN if one is configured. This process works for most client devices; however, it does not work for clients that use an alternate MAC address format. You can configure how MAB authentication is performed for clients with MAC addresses that deviate from the standard format or where the RADIUS configuration requires the user name and password to differ.

If an EAPOL packet is detected on the interface during the lifetime of the link, the switch determines that the device connected to that interface is an 802.1x-capable supplicant and uses 802.1x authentication (not MAC authentication bypass) to authorize the interface. EAPOL history is cleared if the interface link status goes down.

If the switch already authorized a port by using MAC authentication bypass and detects an IEEE 802.1x supplicant, the switch does not unauthorize the client connected to the port. When reauthentication occurs, the switch uses the authentication or reauthentication methods configured on the port, if the previous session ended because the Termination-Action RADIUS attribute value is DEFAULT.

Clients that were authorized with MAC authentication bypass can be reauthenticated. The reauthentication process is the same as that for clients that were authenticated with IEEE 802.1x. During reauthentication, the port remains in the previously assigned VLAN. If reauthentication is successful, the switch keeps the port in the same VLAN. If reauthentication fails, the switch assigns the port to the guest VLAN, if one is configured.

If reauthentication is based on the Session-Timeout RADIUS attribute (Attribute[27]) and the Termination-Action RADIUS attribute (Attribute [29]) and if the Termination-Action RADIUS attribute (Attribute [29]) action is *Initialize* (the attribute value is *DEFAULT*), the MAC authentication bypass session ends, and connectivity is lost during reauthentication. If MAC authentication bypass is enabled and the IEEE 802.1x authentication times out, the switch uses the MAC authentication bypass feature to initiate re-authorization. For more information about these AV pairs, see RFC 3580, “IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines.”

MAC authentication bypass interacts with the features:

- IEEE 802.1x authentication: You can enable MAC authentication bypass only if 802.1x authentication is enabled on the port .
- Guest VLAN: If a client has an invalid MAC address identity, the switch assigns the client to a guest VLAN if one is configured.
- Restricted VLAN: This feature is not supported when the client connected to an IEEE 802.lx port is authenticated with MAC authentication bypass.
- Port security
- Voice VLAN
- Private VLAN: You can assign a client to a private VLAN.
- Network Edge Access Topology (NEAT): MAB and NEAT are mutually exclusive. You cannot enable MAB when NEAT is enabled on an interface, and you should not enable NEAT when MAB is enabled on an interface.

## MAC Authentication Bypass Configuration Guidelines

These are the MAC authentication bypass configuration guidelines:

- Unless otherwise stated, the MAC authentication bypass guidelines are the same as the 802.1x authentication guidelines.

- If you disable MAC authentication bypass from a port after the port has been authorized with its MAC address, the port state is not affected.
- If the port is in the unauthorized state and the client MAC address is not the authentication-server database, the port remains in the unauthorized state. However, if the client MAC address is added to the database, the switch can use MAC authentication bypass to re-authorize the port.
- If the port is in the authorized state, the port remains in this state until re-authorization occurs.
- You can configure a timeout period for hosts that are connected by MAC authentication bypass but are inactive. The range is 1to 65535 seconds.

## 802.1x Multiple Authentication Mode

Multiple-authentication (multiauth) mode allows multiple authenticated clients on the data VLAN and voice VLAN. Each host is individually authenticated. There is no limit to the number of data or voice device that can be authenticated on a multiauthport.

If a hub or access point is connected to an 802.1x-enabled port, each connected client must be authenticated. For non-802.1x devices, you can use MAC authentication bypass or web authentication as the per-host authentication fallback method to authenticate different hosts with different methods on a single port.

You can assign a RADIUS-server-supplied VLAN in multi-auth mode, under the following conditions:

- The host is the first host authorized on the port, and the RADIUS server supplies VLAN information
- Subsequent hosts are authorized with a VLAN that matches the operational VLAN.
- A host is authorized on the port with no VLAN assignment, and subsequent hosts either have no VLAN assignment, or their VLAN information matches the operational VLAN.
- The first host authorized on the port has a group VLAN assignment, and subsequent hosts either have no VLAN assignment, or their group VLAN matches the group VLAN on the port. Subsequent hosts must use the same VLAN from the VLAN group as the first host. If a VLAN list is used, all hosts are subject to the conditions specified in the VLAN list.
- After a VLAN is assigned to a host on the port, subsequent hosts must have matching VLAN information or be denied access to the port.
- The behavior of the critical-auth VLAN is not changed for multi-auth mode. When a host tries to authenticate and the server is not reachable, all authorized hosts are reinitialized in the configured VLAN.

### Multi-auth Per User VLAN assignment

The Multi-auth Per User VLAN assignment feature allows you to create multiple operational access VLANs based on VLANs assigned to the clients on the port that has a single configured access VLAN. The port configured as an access port where the traffic for all the VLANs associated with data domain is not dot1q tagged, and these VLANs are treated as native VLANs.

The number of hosts per multi-auth port is 8, however there can be more hosts.

The following scenarios are associated with the multi-auth Per User VLAN assignments:

**Scenario One**

When a hub is connected to an access port, and the port is configured with an access VLAN (V0).

The host (H1) is assigned to VLAN (V1) through the hub. The operational VLAN of the port is changed to V1. This behaviour is similar on a single-host or multi-domain-auth port.

When a second host (H2) is connected and gets assigned to VLAN ( V2), the port will have two operational VLANs (V1 and V2). If H1 and H2 sends untagged ingress traffic, H1 traffic is mapped to VLAN (V1) and H2 traffic to VLAN (V2), all egress traffic going out of the port on VLAN (V1) and VLAN (V2) are untagged.

If both the hosts, H1 and H2 are logged out or the sessions are removed due to some reason then VLAN (V1) and VLAN (V2) are removed from the port, and the configured VLAN (V0) is restored on the port.

**Scenario Two**

When a hub is connected to an access port, and the port is configured with an access VLAN (V0). The host (H1) is assigned to VLAN (V1) through the hub. The operational VLAN of the port is changed to V1.

When a second host (H2) is connected and gets authorized without explicit vlan policy, H2 is expected to use the configured VLAN (V0) that is restored on the port. A ll egress traffic going out of two operational VLANs, VLAN (V0) and VLAN (V1) are untagged.

If host (H2 ) is logged out or the session is removed due to some reason then the configured VLAN (V0) is removed from the port, and VLAN (V1) becomes the only operational VLAN on the port.

**Scenario Three**

When a hub is connected to an access port in open mode, and the port is configured with an access VLAN (V0) .

The host (H1) is assigned to VLAN (V1) through the hub. The operational VLAN of the port is changed to V1. When a second host (H2) is connected and remains unauthorized, it still has access to operational VLAN (V1) due to open mode.

If host H1 is logged out or the session is removed due to some reason, VLAN (V1) is removed from the port and host (H2) gets assigned to VLAN (V0).

**Note** The combination of Open mode and VLAN assignment has an adverse affect on host (H2) because it has an IP address in the subnet that corresponds to VLAN (V1).

### *Limitation in Multi-Auth Per User VLAN assignment*

In the Multi-Auth Per User VLAN Assignment feature, egress traffic from multiple VLANs are untagged on a port where the hosts receive traffic that is not meant for them. This can be a problem with broadcast and multicast traffic.

- **IPv4 ARPs**: Hosts receive ARP packets from other subnets. This is a problem if two subnets in different Virtual Routing and Forwarding (VRF) tables with overlapping IP address range are active on the port. The host ARP cache may get invalid entries.
- **IPv6 Control Packets**: In IPv6 deployments, Router Advertisements (RA) are processed by hosts that are not supposed to receive them. When a host from one VLAN receives RA from a different VLAN, the host assign incorrect IPv6 address to itself. Such a host is unable to get access to the network.

  The workaround is to enable the IPv6 first hop security so that the broadcast ICMPv6 packets are converted to unicast and sent out from multi-auth enabled ports.. The packet is replicated for each client in multi-auth port belonging to the VLAN and the destination MAC is set to an individual client. Ports having one VLAN, ICMPv6 packets broadcast normally.

- **IP Multicast**: Multicast traffic destined to a multicast group gets replicated for different VLANs if the hosts on those VLANs join the multicast group. When two hosts in different VLANs join a multicast group (on the same mutli-auth port), two copies of each multicast packet are sent out from that port.

## 802.1x Authentication with Guest VLAN

You can configure a guest VLAN for each 802.1x port on the switch to provide limited services to clients, such as downloading the 802.1x client. These clients might be upgrading their system for 802.1x authentication, and some hosts, such as Windows 98 systems, might not be IEEE 802.1x-capable.

When you enable a guest VLAN on an 802.1x port, the switch assigns clients to a guest VLAN when the switch does not receive a response to its EAP request/identity frame or when EAPOL packets are not sent by the client.

The switch maintains the EAPOL packet history. If an EAPOL packet is detected on the interface during the lifetime of the link, the switch determines that the device connected to that interface is an IEEE 802.1x-capable supplicant, and the interface does not change to the guest VLAN state. EAPOL history is cleared if the interface link status goes down. If no EAPOL packet is detected on the interface, the interface changes to the guest VLAN state.

If the switch is trying to authorize an 802.1x-capable voice device and the AAA server is unavailable, the authorization attempt fails, but the detection of the EAPOL packet is saved in the EAPOL history. When the AAA server becomes available, the switch authorizes the voice device. However, the switch no longer allows other devices access to the guest VLAN. To prevent this situation, use one of these command sequences:

- Enter the **authentication event no-response action authorize vlan** *vlan-id* interface configuration command to allow access to the guest VLAN.
- Enter the **shutdown** interface configuration command followed by the **no shutdown** interface configuration command to restart the port.

If devices send EAPOL packets to the switch during the lifetime of the link, the switch no longer allows clients that fail authentication access to the guest VLAN.

**Note** If an EAPOL packet is detected after the interface has changed to the guest VLAN, the interface reverts to an unauthorized state, and 802.1x authentication restarts.

Any number of 802.1x-incapable clients are allowed access when the switch port is moved to the guest VLAN. If an 802.1x-capable client joins the same port on which the guest VLAN is configured, the port is put into the unauthorized state in the user-configured access VLAN, and authentication is restarted.

Guest VLANs are supported on 802.1x ports in single host, multiple host, multi-auth and multi-domain modes.

You can configure any active VLAN except an RSPAN VLAN, a private VLAN, or a voice VLAN as an 802.1x guest VLAN. The guest VLAN feature is not supported on internal VLANs (routed ports) or trunk ports; it is supported only on access ports.

The switch supports *MAC authentication bypass*. When MAC authentication bypass is enabled on an 802.1x port, the switch can authorize clients based on the client MAC address when IEEE 802.1x authentication times out while waiting for an EAPOL message exchange. After detecting a client on an 802.1x port, the switch waits for an Ethernet packet from the client. The switch sends the authentication server a RADIUS-access/request frame with a username and password based on the MAC address. If authorization

succeeds, the switch grants the client access to the network. If authorization fails, the switch assigns the port to the guest VLAN if one is specified.

## 802.1x Authentication with Restricted VLAN

You can configure a restricted VLAN (also referred to as an *authentication failed VLAN*) for each IEEE 802.1x port on a switch stack or a switch to provide limited services to clients that cannot access the guest VLAN. These clients are 802.1x-compliant and cannot access another VLAN because they fail the authentication process. A restricted VLAN allows users without valid credentials in an authentication server (typically, visitors to an enterprise) to access a limited set of services. The administrator can control the services available to the restricted VLAN.

**Note** You can configure a VLAN to be both the guest VLAN and the restricted VLAN if you want to provide the same services to both types of users.

Without this feature, the client attempts and fails authentication indefinitely, and the switch port remains in the spanning-tree blocking state. With this feature, you can configure the switch port to be in the restricted VLAN after a specified number of authentication attempts (the default value is 3 attempts).

The authenticator counts the failed authentication attempts for the client. When this count exceeds the configured maximum number of authentication attempts, the port moves to the restricted VLAN. The failed attempt count increments when the RADIUS server replies with either an *EAP failure* or an empty response without an EAP packet. When the port moves into the restricted VLAN, the failed attempt counter resets.

Users who fail authentication remain in the restricted VLAN until the next reauthentication attempt. A port in the restricted VLAN tries to reauthenticate at configured intervals (the default is 60 seconds). If reauthentication fails, the port remains in the restricted VLAN. If reauthentication is successful, the port moves either to the configured VLAN or to a VLAN sent by the RADIUS server. You can disable reauthentication. If you do this, the only way to restart the authentication process is for the port to receive a *link down* or *EAP logoff* event. We recommend that you keep reauthentication enabled if a client might connect through a hub. When a client disconnects from the hub, the port might not receive the *link down* or *EAP logoff* event.

After a port moves to the restricted VLAN, a simulated EAP success message is sent to the client. This prevents clients from indefinitely attempting authentication. Some clients (for example, devices running Windows XP) cannot implement DHCP without EAP success.

Restricted VLANs are supported on 802.1x ports in all host modes and on Layer 2 ports.

You can configure any active VLAN except an RSPAN VLAN, a primary private VLAN, or a voice VLAN as an 802.1x restricted VLAN. The restricted VLAN feature is not supported on internal VLANs (routed ports) or trunk ports; it is supported only on access ports.

Other security port features such as dynamic ARP Inspection, DHCP snooping, and IP source guard can be configured independently on a restricted VLAN.

## 802.1x Authentication with Inaccessible Authentication Bypass

Use the inaccessible authentication bypass feature, also referred to as *critical authentication* or the *AAA fail policy*, when the switch cannot reach the configured RADIUS servers and new hosts cannot be authenticated. You can configure the switch to connect those hosts to *critical ports.*

When a new host tries to connect to the critical port, that host is moved to a user-specified access VLAN, the *critical VLAN*. The administrator gives limited authentication to the hosts.

When the switch tries to authenticate a host connected to a critical port, the switch checks the status of the configured RADIUS server. If a server is available, the switch can authenticate the host. However, if all the RADIUS servers are unavailable, the switch grants network access to the host and puts the port in the *critical-authentication* state, which is a special case of the authentication state.

**Note** If *critical authentication* is configured on interface, then vlan used for critical authorization (*critical vlan*) should be active on the switch. If the *critical vlan* is inactive (or) down, *critical authentication* session will keep trying to enable inactive vlan and fail repeatedly. This can lead to large amount of memory holding.

### Inaccessible Authentication Bypass Support on Multiple-Authentication Ports

When a port is configured on any host mode and the AAA server is unavailable, the port is then configured to multi-host mode and moved to the critical VLAN. To support this inaccessible bypass on multiple-authentication (multiauth) ports, use the **authentication event server dead action reinitialize vlan** *vlan-id* command. When a new host tries to connect to the critical port, that port is reinitialized and all the connected hosts are moved to the user-specified access VLAN.

This command is supported on all host modes.

### Inaccessible Authentication Bypass Authentication Results

The behavior of the inaccessible authentication bypass feature depends on the authorization state of the port:

- If the port is unauthorized when a host connected to a critical port tries to authenticate and all servers are unavailable, the switch puts the port in the critical-authentication state in the RADIUS-configured or user-specified access VLAN.
- If the port is already authorized and reauthentication occurs, the switch puts the critical port in the critical-authentication state in the current VLAN, which might be the one previously assigned by the RADIUS server.
- If the RADIUS server becomes unavailable during an authentication exchange, the current exchange times out, and the switch puts the critical port in the critical-authentication state during the next authentication attempt.

You can configure the critical port to reinitialize hosts and move them out of the critical VLAN when the RADIUS server is again available. When this is configured, all critical ports in the critical-authentication state are automatically reauthenticated.

### Inaccessible Authentication Bypass Feature Interactions

Inaccessible authentication bypass interacts with these features:

- Guest VLAN: Inaccessible authentication bypass is compatible with guest VLAN. When a guest VLAN is enabled on 8021.x port, the features interact as follows:
  - If at least one RADIUS server is available, the switch assigns a client to a guest VLAN when the switch does not receive a response to its EAP request/identity frame or when EAPOL packets are not sent by the client.
  - If all the RADIUS servers are not available and the client is connected to a critical port, the switch authenticates the client and puts the critical port in the critical-authentication state in the RADIUS-configured or user-specified access VLAN.

    - If all the RADIUS servers are not available and the client is not connected to a critical port, the switch might not assign clients to the guest VLAN if one is configured.
    - If all the RADIUS servers are not available and if a client is connected to a critical port and was previously assigned to a guest VLAN, the switch keeps the port in the guest VLAN.

- Restricted VLAN: If the port is already authorized in a restricted VLAN and the RADIUS servers are unavailable, the switch puts the critical port in the critical-authentication state in the restricted VLAN.
- 802.1x accounting: Accounting is not affected if the RADIUS servers are unavailable.
- Private VLAN: You can configure inaccessible authentication bypass on a private VLAN host port. The access VLAN must be a secondary private VLAN.
- Voice VLAN: Inaccessible authentication bypass is compatible with voice VLAN, but the RADIUS-configured or user-specified access VLAN and the voice VLAN must be different.
- Remote Switched Port Analyzer (RSPAN): Do not configure an RSPAN VLAN as the RADIUS-configured or user-specified access VLAN for inaccessible authentication bypass.

In a switch stack:

When a member switch is added to the stack, the active switch sends the member switch the server status.

## VLAN Assignment, Guest VLAN, Restricted VLAN, and Inaccessible Authentication Bypass

These are the configuration guidelines for VLAN assignment, guest VLAN, restricted VLAN, and inaccessible authentication bypass:

- When 802.1x authentication is enabled on a port, you cannot configure a port VLAN that is equal to a voice VLAN.
- You can configure any VLAN except an RSPAN VLAN or a voice VLAN as an 802.1x guest VLAN. The guest VLAN feature is not supported on internal VLANs (routed ports) or trunk ports; it is supported only on access ports.
- After you configure a guest VLAN for an 802.1x port to which a DHCP client is connected, you might need to get a host IP address from a DHCP server. You can change the settings for restarting the 802.1x authentication process on the switch before the DHCP process on the client times out and tries to get a host IP address from the DHCP server. Decrease the settings for the 802.1x authentication process (**authentication timer inactivity** and **authentication timer reauthentication** interface configuration commands). The amount to decrease the settings depends on the connected 802.1x client type.
- When configuring the inaccessible authentication bypass feature, follow these guidelines:
    - The feature is supported on 802.1x port in single-host mode and multihosts mode.
    - If the client is running Windows XP and the port to which the client is connected is in the critical-authentication state, Windows XP might report that the interface is not authenticated.
    - If the Windows XP client is configured for DHCP and has an IP address from the DHCP server, receiving an EAP-Success message on a critical port might not re-initiate the DHCP configuration process.
    - You can configure the inaccessible authentication bypass feature and the restricted VLAN on an 802.1x port. If the switch tries to reauthenticate a critical port in a restricted VLAN and all the

RADIUS servers are unavailable, switch changes the port state to the critical authentication state and remains in the restricted VLAN.

- You can configure any VLAN except an RSPAN VLAN or a voice VLAN as an 802.1x restricted VLAN. The restricted VLAN feature is not supported on internal VLANs (routed ports) or trunk ports; it is supported only on access ports.

## 802.1x Critical Voice VLAN

When an IP phone connected to a port is authenticated by the Cisco Identity Services Engine (ISE), the phone is put into the voice domain. If the ISE is not reachable, the switch cannot determine if the device is a voice device. If the server is unavailable, the phone cannot access the voice network and therefore cannot operate.

For data traffic, you can configure inaccessible authentication bypass, or critical authentication, to allow traffic to pass through on the native VLAN when the server is not available. If the RADIUS authentication server is unavailable (down) and inaccessible authentication bypass is enabled, the switch grants the client access to the network and puts the port in the critical-authentication state in the RADIUS-configured or the user-specified access VLAN. When the switch cannot reach the configured RADIUS servers and new hosts cannot be authenticated, the switch connects those hosts to critical ports. A new host trying to connect to the critical port is moved to a user-specified access VLAN, the critical VLAN, and granted limited authentication.

**Note** Dynamic assignment of critical voice VLAN is not supported with nested service templates. It causes the device to switch between VLANs continuously in a loop.

You can enter the **authentication event server dead action authorize voice** interface configuration command to configure the critical voice VLAN feature. When the ISE does not respond, the port goes into critical authentication mode. When traffic coming from the host is tagged with the voice VLAN, the connected device (the phone) is put in the configured voice VLAN for the port. The IP phones learn the voice VLAN identification through Cisco Discovery Protocol (Cisco devices) or through LLDP or DHCP.

You can configure the voice VLAN for a port by entering the **switchport voice vlan** *vlan-id* interface configuration command.

This feature is supported in multidomain and multi-auth host modes. Although you can enter the command when the switch in single-host or multi-host mode, the command has no effect unless the device changes to multidomain or multi-auth host mode.

## IEEE 802.1x Authentication with Voice VLAN Ports

A voice VLAN port is a special access port associated with two VLAN identifiers:

- VVID to carry voice traffic to and from the IP phone. The VVID is used to configure the IP phone connected to the port.
- PVID to carry the data traffic to and from the workstation connected to the switch through the IP phone. The PVID is the native VLAN of the port.

The IP phone uses the VVID for its voice traffic, regardless of the authorization state of the port. This allows the phone to work independently of IEEE 802.1x authentication.

In single-host mode, only the IP phone is allowed on the voice VLAN. In multiple-hosts mode, additional clients can send traffic on the voice VLAN after a supplicant is authenticated on the PVID. When multiple-hosts mode is enabled, the supplicant authentication affects both the PVID and the VVID.

A voice VLAN port becomes active when there is a link, and the device MAC address appears after the first Cisco Discovery Protocol message from the IP phone. Cisco IP phones do not relay Cisco Discovery Protocol messages from other devices. As a result, if several IP phones are connected in series, the switch recognizes only the one directly connected to it. When IEEE 802.1x authentication is enabled on a voice VLAN port, the switch drops packets from unrecognized IP phones more than one hop away.

When IEEE 802.1x authentication is enabled on a switch port, you can configure an access port VLAN that is also a voice VLAN.

When IP phones are connected to an 802.1x-enabled switch port that is in single host mode, the switch grants the phones network access without authenticating them. We recommend that you use multidomain authentication (MDA) on the port to authenticate both a data device and a voice device, such as an IP phone

**Note** If you enable IEEE 802.1x authentication on an access port on which a voice VLAN is configured and to which a Cisco IP Phone is connected, the Cisco IP phone loses connectivity to the switch for up to 30 seconds.

## IEEE 802.1x Authentication with Wake-on-LAN

The IEEE 802.1x authentication with wake-on-LAN (WoL) feature allows dormant PCs to be powered when the switch receives a specific Ethernet frame, known as the *magic packet*. You can use this feature in environments where administrators need to connect to systems that have been powered down.

When a host that uses WoL is attached through an IEEE 802.1x port and the host powers off, the IEEE 802.1x port becomes unauthorized. The port can only receive and send EAPOL packets, and WoL magic packets cannot reach the host. When the PC is powered off, it is not authorized, and the switch port is not opened.

When the switch uses IEEE 802.1x authentication with WoL, the switch forwards traffic to unauthorized IEEE 802.1x ports, including magic packets. While the port is unauthorized, the switch continues to block ingress traffic other than EAPOL packets. The host can receive packets but cannot send packets to other devices in the network.

**Note** If PortFast is not enabled on the port, the port is forced to the bidirectional state.

When you configure a port as unidirectional by using the **authentication control-direction in** interface configuration command, the port changes to the spanning-tree forwarding state. The port can send packets to the host but cannot receive packets from the host.

When you configure a port as bidirectional by using the **authentication control-direction both** interface configuration command, the port is access-controlled in both directions. The port does not receive packets from or send packets to the host.

## Flexible Authentication Ordering

You can use flexible authentication ordering to configure the order of methods that a port uses to authenticate a new host. The IEEE 802.1X Flexible Authentication feature supports three authentication methods:

- dot1X: IEEE 802.1X authentication is a Layer 2 authentication method.

- mab: MAC-Authentication Bypass is a Layer 2 authentication method.
- webauth: Web authentication is a Layer 3 authentication method.

Using this feature, you can control which ports use which authentication methods, and you can control the failover sequencing of methods on those ports. For example, MAC authentication bypass and 802.1x can be the primary or secondary authentication methods, and web authentication can be the fallback method if either or both of those authentication attempts fail.

The IEEE 802.1X Flexible Authentication feature supports the following host modes:

- multi-auth: Multiauthentication allows one authentication on a voice VLAN and multiple authentications on the data VLAN.
- multi-domain: Multidomain authentication allows two authentications: one on the voice VLAN and one on the data VLAN.

## Open1x Authentication

Open1x authentication allows a device access to a port before that device is authenticated. When open authentication is configured, a new host can pass traffic according to the access control list (ACL) defined on the port. After the host is authenticated, the policies configured on the RADIUS server are applied to that host.

You can configure open authentication with these scenarios:

- Single-host mode with open authentication–Only one user is allowed network access before and after authentication.
- MDA mode with open authentication–Only one user in the voice domain and one user in the data domain are allowed.
- Multiple-hosts mode with open authentication–Any host can access the network.
- Multiple-authentication mode with open authentication–Similar to MDA, except multiple hosts can be authenticated.

**Note** If open authentication is configured, it takes precedence over other authentication controls. This means that if you use the **authentication open** interface configuration command, the port will grant access to the host irrespective of the **authentication port-control** interface configuration command.

## Multidomain Authentication

The switch supports multidomain authentication (MDA), which allows both a data device and voice device, such as an IP phone, to authenticate on the same switch port. The port is divided into a data domain and a voice domain.

**Note** For all host modes, the line protocol stays up before authorization when port-based authentication is configured.

MDA does not enforce the order of device authentication. However, for best results, we recommend that a voice device is authenticated before a data device on an MDA-enabled port.

Follow these guidelines for configuring MDA:

- You must configure a switch port for MDA.
- You must configure the voice VLAN for the IP phone when the host mode is set to multidomain.
- Voice VLAN assignment on an MDA-enabled port is supported.
- To authorize a voice device, the AAA server must be configured to send a Cisco Attribute-Value (AV) pair attribute with a value of *device-traffic-class=voice*. Without this value, the switch treats the voice device as a data device.

**Note** When traffic-class=voice is downloaded from AAA servers as a service-template, a session will be created in DATA domain instead of VOICE domain.

- The guest VLAN and restricted VLAN features only apply to the data devices on an MDA-enabled port. The switch treats a voice device that fails authorization as a data device.
- If more than one device attempts authorization on either the voice or the data domain of a port, it is error disabled.
- Until a device is authorized, the port drops its traffic. Non-Cisco IP phones or voice devices are allowed into both the data and voice VLANs. The data VLAN allows the voice device to contact a DHCP server to obtain an IP address and acquire the voice VLAN information. After the voice device starts sending on the voice VLAN, its access to the data VLAN is blocked.
- A voice device MAC address that is binding on the data VLAN is not counted towards the port security MAC address limit.
- MDA can use MAC authentication bypass as a fallback mechanism to allow the switch port to connect to devices that do not support IEEE 802.1x authentication.
- When a *data* or a *voice* device is detected on a port, its MAC address is blocked until authorization succeeds. If the authorization fails, the MAC address remains blocked for 5 minutes.
- If more than five devices are detected on the *data* VLAN or more than one voice device is detected on the *voice* VLAN while a port is unauthorized, the port is error disabled.
- When a port host mode is changed from single- or multihost to multidomain mode, an authorized data device remains authorized on the port. However, a Cisco IP phone that has been allowed on the port voice VLAN is automatically removed and must be reauthenticated on that port.
- Active fallback mechanisms such as guest VLAN and restricted VLAN remain configured after a port changes from single- or multihost mode to multidomain mode.
- Switching a port host mode from multidomain to single- or multihost mode removes all authorized devices from the port.
- If a data domain is authorized first and placed in the guest VLAN, non-IEEE 802.1x-capable voice devices need to tag their packets on the voice VLAN to trigger authentication.

- We do not recommend per-user ACLs with an MDA-enabled port. An authorized device with a per-user ACL policy might impact traffic on both the voice and data VLANs of the port. If used, only one device on the port should enforce per-user ACLs.

## 802.1x Supplicant and Authenticator Switches with Network Edge Access Topology

The Network Edge Access Topology (NEAT) feature extends identity to areas outside the wiring closet (such as conference rooms). This allows any type of device to authenticate on the port.

- 802.1x switch supplicant: You can configure a switch to act as a supplicant to another switch by using the 802.1x supplicant feature. This configuration is helpful in a scenario, where, for example, a switch is outside a wiring closet and is connected to an upstream switch through a trunk port. A switch configured with the 802.1x switch supplicant feature authenticates with the upstream switch for secure connectivity. Once the supplicant switch authenticates successfully the port mode changes from access to trunk in an authenticator switch. In a supplicant switch you must manually configure trunk when enabling CISP.
- If the access VLAN is configured on the authenticator switch, it becomes the native VLAN for the trunk port after successful authentication.

In the default state, when you connect a supplicant switch to an authenticator switch that has BPDU guard enabled, the authenticator port could be error-disabled if it receives a Spanning Tree Protocol (STP) bridge protocol data unit (BPDU) packets before the supplicant switch has authenticated. You can control traffic exiting the supplicant port during the authentication period. Entering the **dot1x supplicant controlled transient** global configuration command temporarily blocks the supplicant port during authentication to ensure that the authenticator port does not shut down before authentication completes. If authentication fails, the supplicant port opens. Entering the **no dot1x supplicant controlled transient** global configuration command opens the supplicant port during the authentication period. This is the default behavior.

We strongly recommend using the **dot1x supplicant controlled transient**command on a supplicant switch when BPDU guard is enabled on the authenticator switch port with the **spanning-tree bpduguard enable** interface configuration command.

**Note** If you globally enable BPDU guard on the authenticator switch by using the **spanning-tree portfast bpduguard default** global configuration command, entering the **dot1x supplicant controlled transient** command does not prevent the BPDU violation.

You can enable MDA or multiauth mode on the authenticator switch interface that connects to one more supplicant switches. Multihost mode is not supported on the authenticator switch interface.

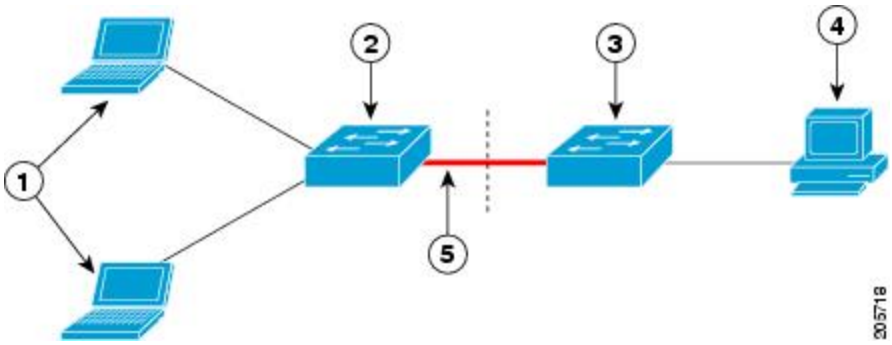
When you reboot an authenticator switch with single-host mode enabled on the interface, the interface may move to err-disabled state before authentication. To recover from err-disabled state, flap the authenticator port to activate the interface again and initiate authentication.

Use the **dot1x supplicant force-multicast** global configuration command on the supplicant switch for Network Edge Access Topology (NEAT) to work in all host modes.

- Host Authorization: Ensures that only traffic from authorized hosts (connecting to the switch with supplicant) is allowed on the network. The switches use Client Information Signalling Protocol (CISP) to send the MAC addresses connecting to the supplicant switch to the authenticator switch.

- Auto enablement: Automatically enables trunk configuration on the authenticator switch, allowing user traffic from multiple VLANs coming from supplicant switches. Configure the cisco-av-pair as *device-traffic-class=switch* at the ISE. (You can configure this under the *group* or the *user* settings.)

***Figure 38: Authenticator and Supplicant Switch using CISP***




| | | | |
|---|---|---|---|
| 1 | Workstations (clients) | 2 | Supplicant switch (outside wiring closet) |
| 3 | Authenticator switch | 4 | Cisco ISE |
| 5 | Trunk port | | |

**Note** The **switchport nonegotiate** command is not supported on supplicant and authenticator switches with NEAT. This command should not be configured at the supplicant side of the topology. If configured on the authenticator side, the internal macros will automatically remove this command from the port.

# 802.1x User Distribution

You can configure 802.1x user distribution to load-balance users with the same group name across multiple different VLANs.

The VLANs are either supplied by the RADIUS server or configured through the switch CLI under a VLAN group name.

- Configure the RADIUS server to send more than one VLAN name for a user. The multiple VLAN names can be sent as part of the response to the user. The 802.1x user distribution tracks all the users in a particular VLAN and achieves load balancing by moving the authorized user to the least populated VLAN.

- Configure the RADIUS server to send a VLAN group name for a user. The VLAN group name can be sent as part of the response to the user. You can search for the selected VLAN group name among the VLAN group names that you configured by using the switch CLI. If the VLAN group name is found, the corresponding VLANs under this VLAN group name are searched to find the least populated VLAN. Load balancing is achieved by moving the corresponding authorized user to that VLAN.

  Whenever the RADIUS server sends a VLAN group name in the attribute as a result of authorization, the least populated VLAN out of the group is assigned to the end-user. In case of reauthentication

(authentication session present) and CoA (session alive), the same VLAN is kept even if it is not the least populated VLAN in the group.

**Note** The RADIUS server can send the VLAN information in any combination of VLAN-IDs, VLAN names, or VLAN groups.

## 802.1x User Distribution Configuration Guidelines

- Confirm that at least one VLAN is mapped to the VLAN group.
- You can map more than one VLAN to a VLAN group.
- You can modify the VLAN group by adding or deleting a VLAN.
- When you clear an existing VLAN from the VLAN group name, none of the authenticated ports in the VLAN are cleared, but the mappings are removed from the existing VLAN group.
- If you clear the last VLAN from the VLAN group name, the VLAN group is cleared.
- You can clear a VLAN group even when the active VLANs are mapped to the group. When you clear a VLAN group, none of the ports or users that are in the authenticated state in any VLAN within the group are cleared, but the VLAN mappings to the VLAN group are cleared.

# Network Admission Control Layer 2 IEEE 802.1x Validation

The switch supports the Network Admission Control (NAC) Layer 2 IEEE 802.1x validation, which checks the antivirus condition or *posture* of endpoint systems or clients before granting the devices network access. With NAC Layer 2 IEEE 802.1x validation, you can do these tasks:

- Download the Session-Timeout RADIUS attribute (Attribute[27]) and the Termination-Action RADIUS attribute (Attribute[29]) from the authentication server.
- Set the number of seconds between reauthentication attempts as the value of the Session-Timeout RADIUS attribute (Attribute[27]) and get an access policy against the client from the RADIUS server.
- Set the action to be taken when the switch tries to reauthenticate the client by using the Termination-Action RADIUS attribute (Attribute[29]). If the value is the *DEFAULT* or is not set, the session ends. If the value is RADIUS-Request, the reauthentication process starts.
- Set the list of VLAN number or name or VLAN group name as the value of the Tunnel Group Private ID (Attribute[81]) and the preference for the VLAN number or name or VLAN group name as the value of the Tunnel Preference (Attribute[83]). If you do not configure the Tunnel Preference, the first Tunnel Group Private ID (Attribute[81]) attribute is picked up from the list.
- View the NAC posture token, which shows the posture of the client, by using the **show authentication** privileged EXEC command.
- Configure secondary private VLANs as guest VLANs.

Configuring NAC Layer 2 IEEE 802.1x validation is similar to configuring IEEE 802.1x port-based authentication except that you must configure a posture token on the RADIUS server.

# Voice Aware 802.1x Security

**Note** To use voice aware IEEE 802.1x authentication, the switch must be running the LAN base image.

You use the voice aware 802.1x security feature to configure the switch to disable only the VLAN on which a security violation occurs, whether it is a data or voice VLAN. In previous releases, when an attempt to authenticate the data client caused a security violation, the entire port shut down, resulting in a complete loss of connectivity.

You can use this feature in IP phone deployments where a PC is connected to the IP phone. A security violation found on the data VLAN results in the shutdown of only the data VLAN. The traffic on the voice VLAN flows through the switch without interruption.

# Common Session ID

Authentication manager uses a single session ID (referred to as a common session ID) for a client no matter which authentication method is used. This ID is used for all reporting purposes, such as the show commands and MIBs. The session ID appears with all per-session syslog messages.

The session ID includes:

- The IP address of the Network Access Device (NAD)
- A monotonically increasing unique 32 bit integer
- The session start time stamp (a 32 bit integer)

This example shows how the session ID appears in the output of the show authentication command. The session ID in this example is 160000050000000B288508E5:

```
Device# show authentication sessions
Interface  MAC Address     Method   Domain   Status         Session ID
Fa4/0/4    0000.0000.0203  mab      DATA     Authz Success  160000050000000B288508E5
```

This is an example of how the session ID appears in the syslog output. The session ID in this example is also160000050000000B288508E5:

```
1w0d: %AUTHMGR-5-START: Starting 'mab' for client (0000.0000.0203) on Interface Fa4/0/4
AuditSessionID 160000050000000B288508E5
1w0d: %MAB-5-SUCCESS: Authentication successful for client (0000.0000.0203) on Interface
Fa4/0/4 AuditSessionID 160000050000000B288508E5
1w0d: %AUTHMGR-7-RESULT: Authentication result 'success' from 'mab' for client
(0000.0000.0203) on Interface Fa4/0/4 AuditSessionID 160000050000000B288508E5
```

The session ID is used by the NAD, the AAA server, and other report-analyzing applications to identify the client. The ID appears automatically. No configuration is required.

# Maximum Number of Allowed Devices Per Port

This is the maximum number of devices allowed on an 802.1x-enabled port:

- In single-host mode, only one device is allowed on the access VLAN. If the port is also configured with a voice VLAN, an unlimited number of Cisco IP phones can send and receive traffic through the voice VLAN.
- In multidomain authentication (MDA) mode, one device is allowed for the access VLAN, and one IP phone is allowed for the voice VLAN.
- In multihost mode, only one 802.1x supplicant is allowed on the port, but an unlimited number of non-802.1x hosts are allowed on the access VLAN. An unlimited number of devices are allowed on the voice VLAN.

# How to Configure IEEE 802.1x Port-Based Authentication

## Configuring 802.1x Authentication

To allow per-user ACLs or VLAN assignment, you must enable AAA authorization to configure the switch for all network-related service requests.

This is the 802.1x AAA process:

### Before you begin

To configure 802.1x port-based authentication, you must enable authentication, authorization, and accounting (AAA) and specify the authentication method list. A method list describes the sequence and authentication method to be queried to authenticate a user.

### Procedure

**Step 1** A user connects to a port on the switch.

**Step 2** Authentication is performed.

**Step 3** VLAN assignment is enabled, as appropriate, based on the RADIUS server configuration.

**Step 4** The switch sends a start message to an accounting server.

**Step 5** Re-authentication is performed, as necessary.

**Step 6** The switch sends an interim accounting update to the accounting server that is based on the result of reauthentication.

**Step 7** The user disconnects from the port.

**Step 8** The switch sends a stop message to the accounting server.

## Configuring 802.1x Port-Based Authentication

Perform these steps to configure 802.1x port-based authentication:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authentication dot1x** {**default**} *method1*<br>**Example:**<br>`Device(config)# aaa authentication dot1x default group radius` | Creates an 802.1x authentication method list.<br>To create a default list that is used when a named list is *not* specified in the **authentication** command, use the **default** keyword followed by the method that is to be used in default situations. The default method list is automatically applied to all ports.<br>For *method1*, enter the **group radius** keywords to use the list of all RADIUS servers for authentication.<br>**Note**<br>Though other keywords are visible in the command-line help string, only the **group radius** keywords are supported. |
| **Step 5** | **dot1x system-auth-control**<br>**Example:**<br>`Device(config)# dot1x system-auth-control` | Enables 802.1x authentication globally on the switch. |
| **Step 6** | **aaa authorization network** {**default**} **group radius**<br>**Example:** | (Optional) Configures the switch to use user-RADIUS authorization for all network-related service requests, such as per-user ACLs or VLAN assignment. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# aaa authorization network default group radius` | |
| **Step 7** | **radius server** *server name*<br>**Example:**<br>`Device(config)# radius server rsim address ipv4 124.2.2.12` | (Optional) Specifies the IP address of the RADIUS server. |
| **Step 8** | **address** {**ipv4** \| **ipv6**} *ip address*<br>**Example:**<br>`Device(config-radius-server)# address ipv4 10.0.1.12` | Configures the IP address for the RADIUS server. |
| **Step 9** | **key** *string*<br>**Example:**<br>`Device(config-radius-server)# key rad123` | (Optional) Specifies the authentication and encryption key used between the switch and the RADIUS daemon running on the RADIUS server. |
| **Step 10** | **exit**<br>**Example:**<br>`Device(config-radius-server)# exit` | Exits the RADIUS server mode and enters the global configuration mode. |
| **Step 11** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/2` | Specifies the port connected to the client that is to be enabled for IEEE 802.1x authentication, and enter interface configuration mode. |
| **Step 12** | **switchport mode access**<br>**Example:**<br>`Device(config-if)# switchport mode access` | (Optional) Sets the port to access mode only if you configured the RADIUS server in Step 6 and Step 7. |
| **Step 13** | **authentication port-control auto**<br>**Example:**<br>`Device(config-if)# authentication` | Enables 802.1x authentication on the port. |

| | Command or Action | Purpose |
|---|---|---|
| | `port-control auto` | |
| **Step 14** | **dot1x pae authenticator**<br>**Example:**<br>`Device(config-if)# dot1x pae authenticator` | Sets the interface Port Access Entity to act only as an authenticator and ignore messages meant for a supplicant. |
| **Step 15** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring Periodic Reauthentication

You can enable periodic 802.1x client reauthentication and specify how often it occurs. If you do not specify a time period before enabling reauthentication, the number of seconds between attempts is 3600.

Follow these steps to enable periodic reauthentication of the client and to configure the number of seconds between reauthentication attempts. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 2/0/1` | Specifies the port to be configured, and enter interface configuration mode. |
| **Step 4** | **authentication periodic**<br>**Example:** | Enables periodic reauthentication of the client, which is disabled by default. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-if)# authentication periodic` | **Note**<br>The default value is 3600 seconds. To change the value of the reauthentication timer or to have the switch use a RADIUS-provided session timeout, enter the **authentication timer reauthenticate** command. |
| **Step 5** | **authentication timer** {{[**inactivity** \| **reauthenticate** \| **restart** \| **unauthorized**]} {*value*}}<br>**Example:**<br>`Device(config-if)# authentication timer reauthenticate 180` | Sets the number of seconds between reauthentication attempts.<br>The **authentication timer** keywords have these meanings:<br>• **inactivity**: Interval in seconds after which if there is no activity from the client then it is unauthorized<br>• **reauthenticate**: Time in seconds after which an automatic reauthentication attempt is initiated<br>• **restart** *value*: Interval in seconds after which an attempt is made to authenticate an unauthorized port<br>• **unauthorized** *value*: Interval in seconds after which an unauthorized session will get deleted<br>This command affects the behavior of the switch only if periodic reauthentication is enabled. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring 802.1x Violation Modes

You can configure an 802.1x port so that it shuts down, generates a syslog error, or discards packets from a new device when:

- A device connects to an 802.1x-enabled port
- The maximum number of allowed about devices have been authenticated on the port

Beginning in privileged EXEC mode, follow these steps to configure the security violation actions on the switch:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authentication dot1x** {**default**} *method1*<br>**Example:**<br>`Device(config)# aaa authentication dot1x default group radius` | Creates an 802.1x authentication method list.<br>To create a default list that is used when a named list is *not* specified in the **authentication** command, use the **default** keyword followed by the method that is to be used in default situations. The default method list is automatically applied to all ports.<br>For *method1*, enter the **group radius** keywords to use the list of all RADIUS servers for authentication. |
| **Step 5** | **interface** *interface-type interface-number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/4` | Specifies the port connected to the client that is to be enabled for IEEE 802.1x authentication, and enter interface configuration mode. |
| **Step 6** | **switchport mode access**<br>**Example:**<br>`Device(config-if)# switchport mode access` | Sets the port to access mode. |
| **Step 7** | **authentication violation** {**shutdown** \| **restrict** \| **protect** \| **replace**} | Configures the violation mode. The keywords have these meanings: |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device(config-if)# authentication violation restrict` | • **shutdown**: Error disable the port.<br>• **restrict**: Generate a syslog error.<br>• **protect**: Drop packets from any new device that sends traffic to the port.<br>• **replace**: Removes the current session and authenticates with the new host. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Changing the Quiet Period

When the switch cannot authenticate the client, the switch remains idle for a set period of time and then tries again. The **authentication timer restart** interface configuration command controls the idle period. A failed authentication of the client might occur because the client provided an invalid password. You can provide a faster response time to the user by entering a number smaller than the default.

Beginning in privileged EXEC mode, follow these steps to change the quiet period. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 2/0/1` | Specifies the port to be configured, and enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **authentication timer restart** *seconds*<br>**Example:**<br>`Device(config-if)# authentication timer restart 30` | Sets the number of seconds that the switch remains in the quiet state following a failed authentication exchange with the client.<br>The range is 1 to 1073741823 seconds; the default is 60. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show authentication sessions interface** *interface-id*<br>**Example:**<br>`Device# show authentication sessions interface gigabitethernet2/0/1` | Verifies your entries. |
| **Step 7** | **copy running-config startup-config**<br>**Example:**<br>`Device# copy running-config startup-config` | (Optional) Saves your entries in the configuration file. |

# Changing the Switch-to-Client Retransmission Time

The client responds to the EAP-request/identity frame from the switch with an EAP-response/identity frame. If the switch does not receive this response, it waits a set period of time (known as the retransmission time) and then resends the frame.

**Note** You should change the default value of this command only to adjust for unusual circumstances such as unreliable links or specific behavioral problems with certain clients and authentication servers.

Beginning in privileged EXEC mode, follow these steps to change the amount of time that the switch waits for client notification. This procedure is optional.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:** | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device> enable` | |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet2/0/1` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 4** | **authentication timer reauthenticate** *seconds*<br>**Example:**<br>`Device(config-if)# authentication timer reauthenticate 60` | Sets the number of seconds that the switch waits for a response to an EAP-request/identity frame from the client before resending the request.<br>The range is 1 to 1073741823 seconds; the default is 5. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show authentication sessions interface** *interface-id*<br>**Example:**<br>`Device# show authentication sessions interface gigabitethernet 2/0/1` | Verifies your entries. |
| **Step 7** | **copy running-config startup-config**<br>**Example:**<br>`Device# copy running-config startup-config` | (Optional) Saves your entries in the configuration file. |

# Setting the Switch-to-Client Frame-Retransmission Number

In addition to changing the switch-to-client retransmission time, you can change the number of times that the switch sends an EAP-request/identity frame (assuming no response is received) to the client before restarting the authentication process.

**Note** You should change the default value of this command only to adjust for unusual circumstances such as unreliable links or specific behavioral problems with certain clients and authentication servers.

Beginning in privileged EXEC mode, follow these steps to set the switch-to-client frame-retransmission number. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device># enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet2/0/1` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 4** | **dot1x max-reauth-req** *count*<br>**Example:**<br>`Device(config-if)# dot1x max-reauth-req 5` | Sets the number of times that the switch sends an EAP-request/identity frame to the client before restarting the authentication process. The range is 1 to 10; the default is 2. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring Host Mode

Beginning in privileged EXEC mode, follow these steps to allow multiple hosts (clients) on an IEEE 802.1x-authorized port that has the **authentication port-control** interface configuration command set to **auto**. Use the **multi-domain** keyword to configure and enable multidomain authentication (MDA), which

allows both a host and a voice device, such as an IP phone (Cisco or non-Cisco), on the same switch port. This procedure is optional.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 2/0/1` | Specifies the port to which multiple hosts are indirectly attached, and enters interface configuration mode. |
| **Step 4** | **authentication host-mode** [**multi-auth** \| **multi-domain** \| **multi-host** \| **single-host**]<br>**Example:**<br>`Device(config-if)# authentication host-mode multi-host` | Allows multiple hosts (clients) on an 802.1x-authorized port.<br>The keywords have these meanings:<br>• **multi-auth**: Allow multiple authenticated clients on both the voice VLAN and data VLAN.<br>**Note**<br>The **multi-auth** keyword is only available with the **authentication host-mode** command.<br>• **multi-host**: Allow multiple hosts on an 802.1x-authorized port after a single host has been authenticated.<br>• **multi-domain**: Allow both a host and a voice device, such as an IP phone (Cisco or non-Cisco), to be authenticated on an IEEE 802.1x-authorized port.<br>**Note**<br>You must configure the voice VLAN for the IP phone when the host mode is set to **multi-domain**. |

| | Command or Action | Purpose |
|---|---|---|
| | | Make sure that the **authentication port-control** interface configuration command is set to **auto** for the specified interface. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Enabling MAC Move

MAC move allows an authenticated host to move from one port on the device to another.

Beginning in privileged EXEC mode, follow these steps to globally enable MAC move on the device. This procedure is optional.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **authentication mac-move permit**<br>**Example:**<br>`Device(config)# authentication mac-move permit` | Enables MAC move on the device. Default is deny.<br>In Session Aware Networking mode, the default CLI is **access-session mac-move deny**. To enable Mac Move in Session Aware Networking, use the **no access-session mac-move** global configuration command.<br>In legacy mode (IBNS 1.0), default value for **mac-move** is **deny** and in C3PL mode (IBNS 2.0) default value is **permit**. |
| **Step 4** | **end**<br>**Example:** | Exits global configuration mode and returns to privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# end` | |

# Disabling MAC Move

Beginning in privileged EXEC mode, follow these steps to disable MAC move from an authentication manager-enabled port to an authentication manager-disabled port on a device. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device# enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **authentication mac-move deny-uncontrolled**<br>**Example:**<br>`Device(config)# authentication mac-move deny-uncontrolled` | Disables MAC move to an authentication manager-disabled port on the device.<br>To disable Mac move in session-aware networking, use the **access-session mac-move deny-uncontrolled** global configuration command. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Returns to privileged EXEC mode. |

# Enabling MAC Replace

MAC replace allows a host to replace an authenticated host on a port.

Beginning in privileged EXEC mode, follow these steps to enable MAC replace on an interface. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Devic(config)# interface gigabitethernet2/0/2` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 4** | **authentication violation** {**protect** \| **replace** \| **restrict** \| **shutdown**}<br>**Example:**<br>`Device(config-if)# authentication violation replace` | Use the **replace** keyword to enable MAC replace on the interface. The port removes the current session and initiates authentication with the new host.<br>The other keywords have these effects:<br>• **protect**: the port drops packets with unexpected MAC addresses without generating a system message.<br>• **restrict**: violating packets are dropped by the CPU and a system message is generated.<br>• **shutdown**: the port is error disabled when it receives an unexpected MAC address. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring 802.1x Accounting

Enabling AAA system accounting with 802.1x accounting allows system reload events to be sent to the accounting RADIUS server for logging. The server can then infer that all active 802.1x sessions are closed.

**Note** Cisco IOS XE Everest 16.6.x, periodic AAA accounting updates are not supported. The switch does not send periodic interim accounting records to the accounting server. Periodic AAA accounting updates are available in Cisco IOS XE Fuji 16.9.x and later releases.

Because RADIUS uses the unreliable UDP transport protocol, accounting messages might be lost due to poor network conditions. If the switch does not receive the accounting response message from the RADIUS server after a configurable number of retransmissions of an accounting request, this system message appears:

```
Accounting message %s for session %s failed to receive Accounting Response.
```

When the stop message is not sent successfully, this message appears:

```
00:09:55: %RADIUS-4-RADIUS_DEAD: RADIUS server 172.20.246.201:1645,1646 is not responding.
```

**Note** You must configure the RADIUS server to perform accounting tasks, such as logging start, stop, and interim-update messages and time stamps. To turn on these functions, enable logging of "Update/Watchdog packets from this AAA client" in your RADIUS server Network Configuration tab. Next, enable "CVS RADIUS Accounting" in your RADIUS server System Configuration tab.

Beginning in privileged EXEC mode, follow these steps to configure 802.1x accounting after AAA is enabled on your switch. This procedure is optional.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/3` | Specifies the port to be configured, and enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **aaa accounting dot1x default start-stop group radius**<br>**Example:**<br>`Device(config-if)# aaa accounting dot1x default start-stop group radius` | Enables 802.1x accounting using the list of all RADIUS servers. |
| **Step 5** | **aaa accounting system default start-stop group radius**<br>**Example:**<br>`Device(config-if)# aaa accounting system default start-stop group radius` | (Optional) Enables system accounting (using the list of all RADIUS servers) and generates system accounting reload event messages when the switch reloads. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring 802.1x Readiness Check

The 802.1x readiness check monitors 802.1x activity on all the switch ports and displays information about the devices connected to the ports that support 802.1x. You can use this feature to determine if the devices connected to the switch ports are 802.1x-capable.

The 802.1x readiness check is allowed on all ports that can be configured for 802.1x. The readiness check is not available on a port that is configured as **dot1x force-unauthorized**.

Follow these steps to enable the 802.1x readiness check on the switch:

### Before you begin

Follow these guidelines to enable the readiness check on the switch:

- The readiness check is typically used before 802.1x is enabled on the switch.
- If you use the **dot1x test eapol-capable** privileged EXEC command without specifying an interface, all the ports on the switch stack are tested.
- When you configure the **dot1x test eapol-capable** command on an 802.1x-enabled port, and the link comes up, the port queries the connected client about its 802.1x capability. When the client responds with a notification packet, it is 802.1x-capable. A syslog message is generated if the client responds within the timeout period. If the client does not respond to the query, the client is not 802.1x-capable. No syslog message is generated.
- The readiness check can be sent on a port that handles multiple hosts (for example, a PC that is connected to an IP phone). A syslog message is generated for each of the clients that respond to the readiness check within the timer period.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **dot1x test eapol-capable** [**interface** *interface-id*]<br>**Example:**<br>`Device# dot1x test eapol-capable interface gigabitethernet1/0/13`<br>`DOT1X_PORT_EAPOL_CAPABLE:DOT1X: MAC 00-01-02-4b-f1-a3 on gigabitethernet1/0/13 is EAPOL capable` | Enables the 802.1x readiness check on the switch.<br>(Optional) For *interface-id* specify the port on which to check for IEEE 802.1x readiness.<br>**Note**<br>If you omit the optional **interface** keyword, all interfaces on the switch are tested. |
| **Step 3** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 4** | **dot1x test timeout** *timeout*<br>**Example:**<br>`Device(config)# dot1x test timeout 54` | (Optional) Configures the timeout used to wait for EAPOL response. The range is from 1 to 65535 seconds. The default is 10 seconds. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring Switch-to-RADIUS Server Communication

Follow these steps to configure the RADIUS server parameters:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:** | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device> enable` | |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip radius source-interface vlan** *vlan interface number*<br>**Example:**<br>`Device(config)# ip radius source-interface vlan 80` | Specifies that the RADIUS packets have the IP address of the indicated interface. |
| **Step 4** | **radius server** *server name*<br>**Example:**<br>`Device(config)# radius server rsim address ipv4 172.16.0.1` | (Optional) Specifies the IP address of the RADIUS server. |
| **Step 5** | **address** {**ipv4** \| **ipv6**} *ip address*<br>**Example:**<br>`Device(config-radius-server)# address ipv4 10.0.1.2 auth-port 1550 acct-port 1560` | Configures the IP address for the RADIUS server. |
| **Step 6** | **key** *string*<br>**Example:**<br>`Device(config-radius-server)# key rad123` | (Optional) Specifies the authentication and encryption key used between the switch and the RADIUS daemon running on the RADIUS server. |
| **Step 7** | **exit**<br>**Example:**<br>`Device(config-radius-server)# exit` | Exits the RADIUS server mode and enters the global configuration mode. |
| **Step 8** | **radius-server dead-criteria tries** *num-tries*<br>**Example:**<br>`Device(config)# radius-server` | Specifies the number of unanswered sent messages to a RADIUS server before considering the server to be inactive. The range of *num-tries* is 1 to 100. |

| | Command or Action | Purpose |
|---|---|---|
| | `dead-criteria tries 30` | |
| **Step 9** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Setting the Reauthentication Number

You can also change the number of times that the device restarts the authentication process before the port changes to the unauthorized state.

**Note** You should change the default value of this command only to adjust for unusual circumstances such as unreliable links or specific behavioral problems with certain clients and authentication servers.

Beginning in privileged EXEC mode, follow these steps to set the reauthentication number. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device# interface gigabitethernet2/0/1` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 4** | **switchport mode access**<br>**Example:** | Sets the port to access mode only if you previously configured the RADIUS server. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-if)# switchport mode access` | |
| **Step 5** | **dot1x max-req** *count*<br>**Example:**<br>`Device(config-if)# dot1x max-req 4` | Sets the number of times that the device restarts the authentication process before the port changes to the unauthorized state. The range is 0 to 10; the default is 2. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring a Guest VLAN

When you configure a guest VLAN, clients that are not 802.1x-capable are put into the guest VLAN when the server does not receive a response to its EAP request/identity frame. Clients that are 802.1x-capable but that fail authentication are not granted network access. The switch supports guest VLANs in single-host or multiple-hosts mode.

Beginning in privileged EXEC mode, follow these steps to configure a guest VLAN. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 2/0/2` | Specifies the port to be configured, and enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **dot1x port-control auto**<br>**Example:**<br>`Device(config-if)# dot1x port-control auto` | Enables 802.1x authentication on the port. |
| **Step 5** | **authentication event no-response action authorize vlan** *vlan-id*<br>**Example:**<br>`Device(config-if)# authentication event no-response action authorize vlan 2` | Specifies an active VLAN as an 802.1x guest VLAN. The range is 1 to 4094.<br>You can configure any active VLAN except an internal VLAN (routed port), an RSPAN VLAN or a voice VLAN as an 802.1x guest VLAN. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring a Restricted VLAN

When you configure a restricted VLAN on a device, clients that are IEEE 802.1x-compliant are moved into the restricted VLAN when the authentication server does not receive a valid username and password. The device supports restricted VLANs only in single-host mode.

Beginning in privileged EXEC mode, follow these steps to configure a restricted VLAN. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:** | Specifies the port to be configured, and enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# interface gigabitethernet 2/0/2` | |
| **Step 4** | **authentication port-control auto**<br>**Example:**<br>`Device(config-if)# authentication port-control auto` | Enables 802.1x authentication on the port. |
| **Step 5** | **authentication event fail action authorize vlan** *vlan-id*<br>**Example:**<br>`Device(config-if)# authentication event fail action authorize vlan 2` | Specifies an active VLAN as an 802.1x restricted VLAN. The range is 1 to 4094.<br>You can configure any active VLAN except an internal VLAN (routed port), an RSPAN VLAN or a voice VLAN as an 802.1x restricted VLAN. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

## Configuring the Number of Authentication Attempts on a Restricted VLAN

You can configure the maximum number of authentication attempts allowed before a user is assigned to the restricted VLAN by using the **authentication event fail retry** *retry count* interface configuration command. The range of allowable authentication attempts is 1 to 3. The default is 3 attempts.

Beginning in privileged EXEC mode, follow these steps to configure the maximum number of allowed authentication attempts. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 2/0/3` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 4** | **authentication port-control auto**<br>**Example:**<br>`Device(config-if)# authentication port-control auto` | Enables 802.1x authentication on the port. |
| **Step 5** | **authentication event fail action authorize vlan** *vlan-id*<br>**Example:**<br>`Device(config-if)# authentication event fail action authorize vlan 8` | Specifies an active VLAN as an 802.1x restricted VLAN. The range is 1 to 4094.<br>You can configure any active VLAN except an internal VLAN (routed port), an RSPAN VLAN or a voice VLAN as an 802.1x restricted VLAN. |
| **Step 6** | **authentication event fail retry** *retry count*<br>**Example:**<br>`Device(config-if)# authentication event fail retry 2` | Specifies a number of authentication attempts before a port moves to the auth fail VLAN. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring 802.1x Inaccessible Authentication Bypass with Critical Voice VLAN

Beginning in privileged EXEC mode, follow these steps to configure critical voice VLAN on a port and enable the inaccessible authentication bypass feature.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:** | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device> enable` | |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **radius-server dead-criteria** {**time** *seconds* } [**tries** *number*]<br>**Example:**<br>`Device(config)# radius-server dead-criteria time 20 tries 10` | Sets the conditions that determine when a RADIUS server is considered un-available or down (dead).<br>• **time**: 1 to 120 seconds. The switch dynamically determines a default *seconds* value between 10 and 60.<br>• **number**: 1 to 100 tries. The switch dynamically determines a default **tries***number* between 10 and 100. |
| **Step 5** | **radius-serverdeadtime***minutes*<br>**Example:**<br>`Device(config)# radius-server deadtime 60` | (Optional) Sets the number of minutes during which a RADIUS server is not sent requests. The range is from 0 to 1440 minutes (24 hours). The default is 0 minutes. |
| **Step 6** | **radius server** *server name*<br>**Example:**<br>`Device(config)# radius server rsim address ipv4 124.2.2.12` | (Optional) Specifies the IP address of the RADIUS server. |
| **Step 7** | **address** {**ipv4 \| ipv6**} *ip address* **auth-port** *port_number* **acct-port** *port_number*<br>**Example:**<br>`Device(config-radius-server)# address ipv4 10.0.1.2 auth-port 1550 acct-port 1560` | Configures the IP address for the RADIUS server. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **key** *string*<br>**Example:**<br>`Device(config-radius-server)# key rad123` | (Optional) Specifies the authentication and encryption key used between the switch and the RADIUS daemon running on the RADIUS server. |
| **Step 9** | **exit**<br>**Example:**<br>`Device(config-radius-server)# exit` | Exits the RADIUS server mode and enters the global configuration mode. |
| **Step 10** | **dot1x critical** {**eapol** \| **recovery delay** *milliseconds*}<br>**Example:**<br>`Device(config)# dot1x critical eapol`<br>`Device(config)# dot1x critical recovery delay 2000` | (Optional) Configure the parameters for inaccessible authentication bypass:<br>• **eapol**: Specify that the switch sends an EAPOL-Success message when the switch successfully authenticates the critical port.<br>• **recovery delay***milliseconds*: Set the recovery delay period during which the switch waits to re-initialize a critical port when a RADIUS server that was unavailable becomes available. The range is from 1 to 10000 milliseconds. The default is 1000 milliseconds (a port can be re-initialized every second). |
| **Step 11** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Specify the port to be configured, and enters interface configuration mode. |
| **Step 12** | **authentication event server dead action** {**authorize** \| **reinitialize**} **vlan** *vlan-id*]<br>**Example:**<br>`Device(config-if)# authentication event server dead action reinitialicze vlan 20` | Use these keywords to move hosts on the port if the RADIUS server is unreachable:<br>• **authorize**: Move any new hosts trying to authenticate to the user-specified critical VLAN.<br>• **reinitialize**: Move all authorized hosts on the port to the user-specified critical VLAN. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 13** | **switchport voice vlan** *vlan-id*<br>**Example:**<br>`Device(config-if)# switchport voice vlan` | Specifies the voice VLAN for the port. The voice VLAN cannot be the same as the critical data VLAN configured in Step 6. |
| **Step 14** | **authentication event server dead action authorize voice**<br>**Example:**<br>`Device(config-if)# authentication event server dead action authorize voice` | Configures critical voice VLAN to move data traffic on the port to the voice VLAN if the RADIUS server is unreachable. |
| **Step 15** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 16** | **show authentication interface** *interface-id*<br>**Example:**<br>`Device(config-if)# show authentication interface gigabitethernet 1/0/1` | (Optional) Verify your entries. |

### Example

To return to the RADIUS server default settings, use the **no radius-server dead-criteria**, the **no radius-server deadtime**, and the **no radius server** global configuration commands. To disable inaccessible authentication bypass, use the **no authentication event server dead action** interface configuration command. To disable critical voice VLAN, use the no **authentication event server dead action authorize voice** interface configuration command.

# Configuring 802.1x Authentication with Wake-on-LAN

Beginning in privileged EXEC mode, follow these steps to enable 802.1x authentication with wake-on-LAN (WoL). This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** | Enables privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device> enable` | • Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet2/0/3` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 4** | **authentication control-direction** {**both** \| **in**}<br>**Example:**<br>`Device(config-if)# authentication control-direction both` | Enables 802.1x authentication with WoL on the port, and use these keywords to configure the port as bidirectional or unidirectional.<br>• **both**: Sets the port as bidirectional. The port cannot receive packets from or send packets to the host. By default, the port is bidirectional.<br>• **in**: Sets the port as unidirectional. The port can send packets to the host but cannot receive packets from the host. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show authentication sessions interface** *interface-id*<br>**Example:**<br>`Device# show authentication sessions interface gigabitethernet2/0/3` | Displays information about current Auth Manager sessions on the interface. |

# Configuring MAC Authentication Bypass

Beginning in privileged EXEC mode, follow these steps to enable MAC authentication bypass. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 2/0/1` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 4** | **authentication port-control auto**<br>**Example:**<br>`Device(config-if)# authentication port-control auto` | Enables 802.1x authentication on the port. |
| **Step 5** | **mab** [**eap**]<br>**Example:**<br>`Device(config-if)# mab` | Enables MAC authentication bypass.<br>(Optional) Use the **eap** keyword to configure the device to use EAP for authorization. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring 802.1x User Distribution

Beginning in privileged EXEC mode, follow these steps to configure a VLAN group and to map a VLAN to it:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **vlan group** *vlan-group-name* **vlan-list** *vlan-list*<br>**Example:**<br>`Device(config)# vlan group eng-dept vlan-list 10` | Configures a VLAN group, and maps a single VLAN or a range of VLANs to it. |
| **Step 4** | **no vlan group** *vlan-group-name* **vlan-list** *vlan-list*<br>**Example:**<br>`Device(config)# no vlan group eng-dept vlan-list 10` | Clears the VLAN group configuration or elements of the VLAN group configuration. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring NAC Layer 2 802.1x Validation

You can configure NAC Layer 2 802.1x validation, which is also referred to as 802.1x authentication with a RADIUS server.

Beginning in privileged EXEC mode, follow these steps to configure NAC Layer 2 802.1x validation. The procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** | Enables privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device> enable` | • Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet2/0/3` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 4** | **switchport mode access**<br>**Example:**<br>`Device(config-if)# switchport mode access` | Sets the port to access mode only if you configured the RADIUS server. |
| **Step 5** | **authentication event no-response action authorize vlan** *vlan-id*<br>**Example:**<br>`Device(config-if)# authentication event no-response action authorize vlan 8` | Specifies an active VLAN as an 802.1x guest VLAN. The range is 1 to 4094.<br>You can configure any active VLAN except an internal VLAN (routed port), an RSPAN VLAN, or a voice VLAN as an 802.1x guest VLAN. |
| **Step 6** | **authentication periodic**<br>**Example:**<br>`Device(config-if)# authentication periodic` | Enables periodic reauthentication of the client, which is disabled by default. |
| **Step 7** | **authentication timer reauthenticate**<br>**Example:**<br>`Device(config-if)# authentication timer reauthenticate` | Sets reauthentication attempt for the client (set to one hour).<br>This command affects the behavior of the switch only if periodic reauthentication is enabled. |
| **Step 8** | **end**<br>**Example:** | Exits interface configuration mode and returns to privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-if)# end` | |
| **Step 9** | **show authentication sessions interface** *interface-id*<br>**Example:**<br>`Device# show authentication sessions interface gigabitethernet2/0/3` | Displays information about current Auth Manager sessions on the interface. |

# Configuring an Authenticator Switch with NEAT

Configuring this feature requires that one switch outside a wiring closet is configured as a supplicant and is connected to an authenticator switch.

**Note**

- The authenticator switch interface configuration must be restored to access mode by explicitly flapping it if a line card is removed and inserted in the chassis when CISP or NEAT session is active.
- The *cisco-av-pairs* must be configured as *device-traffic-class=switch* on the ISE, which sets the interface as a trunk after the supplicant is successfully authenticated.

Beginning in privileged EXEC mode, follow these steps to configure a switch as an authenticator:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **cisp enable**<br>**Example:**<br>`Device(config)# cisp enable` | Enables CISP. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 2/0/1` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 5** | **switchport mode access**<br>**Example:**<br>`Device(config-if)# switchport mode access` | Sets the port mode to **access**. |
| **Step 6** | **authentication port-control auto**<br>**Example:**<br>`Device(config-if)# authentication port-control auto` | Sets the port-authentication mode to auto. |
| **Step 7** | **dot1x pae authenticator**<br>**Example:**<br>`Device(config-if)# dot1x pae authenticator` | Configures the interface as a port access entity (PAE) authenticator. |
| **Step 8** | **spanning-tree portfast**<br>**Example:**<br>`Device(config-if)# spanning-tree portfast trunk` | Enables Port Fast on an access port connected to a single workstation or server.. |
| **Step 9** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring a Supplicant Switch with NEAT

Beginning in privileged EXEC mode, follow these steps to configure a switch as a supplicant:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **cisp enable**<br>**Example:**<br>`Device(config)# cisp enable` | Enables CISP. |
| **Step 4** | **dot1x credential**s *profile*<br>**Example:**<br>`Device(config)# dot1x credentials test` | Creates 802.1x credentials profile. This must be attached to the port that is configured as supplicant. |
| **Step 5** | **username** *suppswitch*<br>**Example:**<br>`Device(config)# username suppswitch` | Creates a username. |
| **Step 6** | **password** *password*<br>**Example:**<br>`Device(config)# password myswitch` | Creates a password for the new username. |
| **Step 7** | **dot1x supplicant force-multicast**<br>**Example:**<br>`Device(config)# dot1x supplicant force-multicast` | Forces the switch to send only multicast EAPOL packets when it receives either unicast or multicast packets.<br>This also allows NEAT to work on the supplicant switch in all host modes. |
| **Step 8** | **interface** *interface-id*<br>**Example:** | Specifies the port to be configured, and enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# interface gigabitethernet1/0/1` | |
| **Step 9** | **switchport trunk encapsulation dot1q**<br>**Example:**<br>`Device(config-if)# switchport trunk encapsulation dot1q` | Sets the port to trunk mode. |
| **Step 10** | **switchport mode trunk**<br>**Example:**<br>`Device(config-if)# switchport mode trunk` | Configures the interface as a VLAN trunk port. |
| **Step 11** | **dot1x pae supplicant**<br>**Example:**<br>`Device(config-if)# dot1x pae supplicant` | Configures the interface as a port access entity (PAE) supplicant. |
| **Step 12** | **dot1x credentials** *profile-name*<br>**Example:**<br>`Device(config-if)# dot1x credentials test` | Attaches the 802.1x credentials profile to the interface. |
| **Step 13** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring 802.1x Authentication with Downloadable ACLs and Redirect URLs

**Note** You must configure a downloadable ACL on the ACS before downloading it to the switch.

After authentication on the port, you can use the **show ip access-list** privileged EXEC command to display the downloaded ACLs on the port.

**Note** The output of the **show ip access-lists interface** command does not display dACL or ACL filter IDs. This is because the ACLs are attached to the virtual ports created by multidomain authentication for each authentication session; instead of the physical interface. To display dACL or ACL filter IDs, use the **show ip access-lists** *access-list-name* command. The *access-list-name* should be taken from the **show access-session interface** *interface-name* **detail** command output. The *access-list-name* is case sensitive.

## Configuring Downloadable ACLs

The policies take effect after client authentication and the client IP address addition to the IP device tracking table. The switch then applies the downloadable ACL to the port.

Beginning in privileged EXEC mode:

### Before you begin

SISF-Based device tracking is a prerequisite to configuring 802.1x authentication. Ensure that you have enabled device tracking programmatically or manually. For more information, see the *Confguring SISF-Based Tracking* chapter.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authorization network default local group radius**<br>**Example:**<br>`Device(config)# aaa authorization network default local group radius` | Sets the authorization method to local. To remove the authorization method, use the **no aaa authorization network default local group radius** command. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **radius-server vsa send authentication**<br>**Example:**<br>`Device(config)# radius-server vsa send authentication` | Configures the radius vsa send authentication. |
| **Step 6** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet2/0/4` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 7** | **ip access-group** *acl-id* **in**<br>**Example:**<br>`Device(config-if)# ip access-group default_acl in` | Configures the default ACL on the port in the input direction.<br>**Note**<br>The *acl-id* is an access list name or number. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

## Configuring a Downloadable Policy

### Before you begin

SISF-Based device tracking is a prerequisite to configuring 802.1x authentication. Ensure that you have enabled device tracking programmatically or manually.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | **access-list** *access-list-number* **{ deny \| permit } { hostname \| any \| host } log**<br>**Example:**<br>`Device(config)# access-list 1 deny any log` | Defines the default port ACL.<br>The access-list-number is a decimal number from 1 to 99 or 1300 to 1999.<br>Enter **deny** or **permit** to specify whether to deny or permit access if conditions are matched.<br>The source is the source address of the network or host that sends a packet, such as this:<br>• **hostname**: The 32-bit quantity in dotted-decimal format.<br>• **any**: The keyword any as an abbreviation for source and source-wildcard value of 0.0.0.0 255.255.255.255. You do not need to enter a source-wildcard value.<br>• **host**: The keyword host as an abbreviation for source and source-wildcard of source 0.0.0.0.<br>(Optional) Applies the source-wildcard wildcard bits to the source.<br>(Optional) Enters log to cause an informational logging message about the packet that matches the entry to be sent to the console. |
| **Step 4** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 2/0/2` | Enters interface configuration mode. |
| **Step 5** | **ip access-group** *acl-id* **in**<br>**Example:**<br>`Device(config-if)# ip access-group default_acl in` | Configures the default ACL on the port in the input direction.<br>**Note**<br>The *acl-id* is an access list name or number. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(config-if)# exit` | Exits interface configuration mode and returns to global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 8** | **aaa authorization network default group radius**<br>**Example:**<br>`Device(config)# aaa authorization network default group radius` | Sets the authorization method to local. To remove the authorization method, use the **no aaa authorization network default group radius** command. |
| **Step 9** | **radius-server vsa send authentication**<br>**Example:**<br>`Device(config)# radius-server vsa send authentication` | Configures the network access server to recognize and use vendor-specific attributes.<br>**Note**<br>The downloadable ACL must be operational. |
| **Step 10** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring VLAN ID Based MAC Authentication

Beginning in privileged EXEC mode, follow these steps:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **mab request format attribute 32 vlan access-vlan**<br>**Example:**<br>`Device(config)# mab request format attribute 32 vlan access-vlan` | Enables VLAN ID-based MAC authentication. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring Flexible Authentication Ordering

The examples used in the instructions below changes the order of Flexible Authentication Ordering so that MAB is attempted before IEEE 802.1X authentication (dot1x). MAB is configured as the first authentication method, so MAB will have priority over all other authentication methods.

Beginning in privileged EXEC mode, follow these steps:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 4** | **switchport mode access**<br>**Example:** | Sets the port to access mode only if you previously configured the RADIUS server. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-if)# switchport mode access` | |
| **Step 5** | **authentication order** [ **dot1x** \| **mab** ] \| {**webauth**}<br>**Example:**<br>`Device(config-if)# authentication order mab dot1x` | (Optional) Sets the order of authentication methods used on a port. |
| **Step 6** | **authentication priority** [ **dot1x** \| **mab** ] \| {**webauth**}<br>**Example:**<br>`Device(config-if)# authentication priority mab dot1x` | (Optional) Adds an authentication method to the port-priority list. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring Open1x

Beginning in privileged EXEC mode, follow these steps to enable manual control of the port authorization state:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Specifies the port to be configured, and enters interface configuration mode. |
| **Step 4** | **switchport mode access**<br>**Example:**<br>`Device(config-if)# switchport mode access` | Sets the port to access mode only if you configured the RADIUS server. |
| **Step 5** | **authentication control-direction** {**both** \| **in**}<br>**Example:**<br>`Device(config-if)# authentication control-direction both` | (Optional) Configures the port control as unidirectional or bidirectional. |
| **Step 6** | **authentication fallback** *name*<br>**Example:**<br>`Device(config-if)# authentication fallback profile1` | (Optional) Configures a port to use web authentication as a fallback method for clients that do not support 802.1x authentication. |
| **Step 7** | **authentication host-mode** [**multi-auth** \| **multi-domain** \| **multi-host** \| **single-host**]<br>**Example:**<br>`Device(config-if)# authentication host-mode multi-auth` | (Optional) Sets the authorization manager mode on a port. |
| **Step 8** | **authentication open**<br>**Example:**<br>`Device(config-if)# authentication open` | (Optional) Enables or disable open access on a port. |
| **Step 9** | **authentication order** [ **dot1x** \| **mab** ] \| {**webauth**}<br>**Example:**<br>`Device(config-if)# authentication order dot1x webauth` | (Optional) Sets the order of authentication methods used on a port. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 10** | **authentication periodic**<br>**Example:**<br>`Device(config-if)# authentication periodic` | (Optional) Enables or disable reauthentication on a port. |
| **Step 11** | **authentication port-control** {**auto** \| **force-authorized** \| **force-un authorized**}<br>**Example:**<br>`Device(config-if)# authentication port-control auto` | (Optional) Enables manual control of the port authorization state. |
| **Step 12** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Disabling 802.1x Authentication on a Port

You can disable 802.1x authentication on the port by using the **no dot1x pae** interface configuration command.

Beginning in privileged EXEC mode, follow these steps to disable 802.1x authentication on the port. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet` | Specifies the port to be configured, and enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `2/0/1` | |
| **Step 4** | **switchport mode access**<br>**Example:**<br>`Device(config-if)# switchport mode access` | (Optional) Sets the port to access mode only if you configured the RADIUS server. |
| **Step 5** | **no dot1x pae authenticator**<br>**Example:**<br>`Device(config-if)# no dot1x pae authenticator` | Disables 802.1x authentication on the port. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Resetting the 802.1x Authentication Configuration to Default Values

Beginning in privileged EXEC mode, follow these steps to reset the 802.1x authentication configuration to the default values. This procedure is optional.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/2` | Enters interface configuration mode, and specifies the port to be configured. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **dot1x default**<br>**Example:**<br>`Device(config-if)# dot1x default` | Resets the 802.1x parameters to the default values. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Configuring Voice-Aware 802.1x Security

You use the voice aware 802.1x security feature on the device to disable only the VLAN on which a security violation occurs, whether it is a data or voice VLAN. You can use this feature in IP phone deployments where a PC is connected to the IP phone. A security violation found on the data VLAN results in the shutdown of only the data VLAN. The traffic on the voice VLAN flows through the device without interruption.

Follow these guidelines to configure voice aware 802.1x voice security on the device:

- You enable voice aware 802.1x security by entering the **errdisable detect cause security-violation shutdown vlan** global configuration command. You disable voice aware 802.1x security by entering the **no** version of this command. This command applies to all 802.1x-configured ports in the device.

**Note** If you do not include the **shutdown vlan** keywords, the entire port is shut down when it enters the error-disabled state.

- If you use the **errdisable recovery cause security-violation** global configuration command to configure error-disabled recovery, the port is automatically re-enabled. If error-disabled recovery is not configured for the port, you re-enable it by using the **shutdown** and **no shutdown** interface configuration commands.
- You can re-enable individual VLANs by using the **clear errdisable interface** *interface-id* **vlan** [*vlan-list*] privileged EXEC command. If you do not specify a range, all VLANs on the port are enabled.

Follow these steps to enable voice aware 802.1x security:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **errdisable detect cause security-violation shutdown vlan**<br>**Example:**<br>`Device(config)# errdisable detect cause security-violation shutdown vlan` | Shut down any VLAN on which a security violation error occurs.<br>**Note**<br>If the **shutdown vlan** keywords are not included, the entire port enters the error-disabled state and shuts down. |
| **Step 4** | **errdisable recovery cause security-violation**<br>**Example:**<br>`Device(config)# errdisable recovery cause security-violation` | Enables the automatic recovery of ports that were disabled because of 802.1X security violations.. |
| **Step 5** | Enter the following:<br>• **shutdown**<br>• **no shutdown**<br>**Example:**<br>`Device(config)# no shutdown` | (Optional) Re-enables an error-disabled VLAN, and clear all error-disable indications. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and returns to privileged EXEC mode. |
| **Step 7** | **clear errdisable interface***interface-id* **vlan** *[vlan-list]*<br>**Example:**<br>`Device# clear errdisable interface gigabitethernet 0/1/1 vlan vlan_list` | (Optional) Reenables individual VLANs that have been error disabled.<br>• For the *interface-id* argument, specify the port on which to reenable individual VLANs.<br>• (Optional) For the *vlan-list* argument, specify a list of VLANs to be re-enabled. If vlan-list is not specified, all VLANs are re-enabled. |
| **Step 8** | **show errdisable detect**<br>**Example:** | Displays the error-disabled detection status. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# show errdisable detect` | |

# Configuration Examples for IEEE 802.1x Port-Based Authentication

The following sections provide configuration examples for IEEE 802.1x port-based authentication.

## Example: Configuring Inaccessible Authentication Bypass

This example shows how to configure the Inaccessible Authentication Bypass feature:

```
Device> enable
Device# configure terminal
Device(config)# radius-server dead-criteria time 30 tries 20
Device(config)# radius-server deadtime 60
Device(config)# radius server server1
Device(config-radius-server)# address ipv4 172.29.36.49 acct-port 1618 auth-port 1612
Device(config-radius-server)# key abc1234
Device(config-radius-server)# exit
Device(config)# dot1x critical eapol
Device(config)# dot1x critical recovery delay 2000
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# dot1x critical
Device(config-if)# dot1x critical recovery action reinitialize
Device(config-if)# dot1x critical vlan 20
Device(config-if)# end
```

## Example: Configuring VLAN Groups

This example shows how to configure VLAN groups, map VLANs to groups, and verify VLAN group configurations and mappings to specified VLANs:

```
Device> enable
Device(config)# vlan group eng-dept vlan-list 10
Device(config)# exit
Device# show vlan group group-name eng-dept

Group Name                      Vlans Mapped
-------------                   --------------
eng-dept                        10

Device# show dot1x vlan-group all

Group Name                      Vlans Mapped
-------------                   --------------
eng-dept                        10
hr-dept                         20
```

This example shows how to add a VLAN to an existing VLAN group and to verify that the VLAN is added:

```
Device> enable
Device(config)# vlan group eng-dept vlan-list 30
Device(config)# exit
Device(config)# show vlan group eng-dept

Group Name                      Vlans Mapped
-------------                   --------------
eng-dept                        10,30
```

This example shows how to remove a VLAN from a VLAN group:

```
Device> enable
Device# no vlan group eng-dept vlan-list 10
```

This example shows that when all the VLANs are cleared from a VLAN group, the VLAN group is cleared:

```
Device> enable
Device(config)# no vlan group eng-dept vlan-list 30
Vlan 30 is successfully cleared from vlan group eng-dept.
Device(config)# exit
Device# show vlan group group-name eng-dept
```

This example shows how to clear all the VLAN groups:

```
Device> enable
Device(config)# no vlan group end-dept vlan-list all
Device(config)# exit
Device# show vlan-group all
```

# Monitoring IEEE 802.1x Port-Based Authentication Statistics and Status

This section list the commands to monitor IEEE 802.1x port-based authentication statistics and status.

***Table 29: Privileged EXEC show Commands***

| Command | Purpose |
|---|---|
| **show dot1x all statistics** | Displays 802.1x statistics for all ports |
| **show dot1x interface** *interface-id* **statistics** | Displays 802.1x statistics for a specific port |
| **show dot1x all** [**count** \| **details** \| **statistics** \| **summary**] | Displays the 802.1x administrative and operational status for a switch |
| **show dot1x interface** *interface-id* | Displays the 802.1x administrative and operational status for a specific port |

*Table 30: Global Configuration Commands*

| Command | Purpose |
|---|---|
| **no dot1x logging verbose** | Filters verbose 802.1x authentication messages. |

# Feature History for IEEE 802.1x Port-Based Authentication

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | IEEE 802.1x Port-Based Authentication | IEEE 802.1x authentication prevents unauthorized devices (clients) from gaining access to the network.<br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | IEEE 802.1x Port-Based Authentication | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Amsterdam 17.2.1 | Mac Move | MAC move from an authentication manager-enabled port to an authentication manager-disabled port on a device is now enabled by default. |
| Cisco IOS XE Amsterdam 17.2.1 | Session Limit - To prevent MAC address flooding DOS attack | The **access-session limit profile** command was introduced to limit the number of voice and data hosts connecting to a port. |
| Cisco IOS XE Bengaluru 17.5.1 | Session Timer Support - Authentication Timer Reauthenticate | The authentication timer reauthenticate supported time-out range was increased from 1 to 65535 seconds to 1 to 1073741823 seconds. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 29

# IEEE 802.1X VLAN Assignment

The IEEE 802.1X VLAN Assignment feature is automatically enabled when IEEE 802.1X authentication is configured for an access port, which allows the RADIUS server to send a VLAN assignment to the device port. This assignment configures the device port so that network access can be limited for certain users.


- Prerequisites for IEEE 802.1X VLAN Assignment, on page 595
- Restrictions for IEEE 802.1X VLAN Assignment, on page 596
- Information About IEEE 802.1X VLAN Assignment, on page 597
- How to Configure IEEE 802.1X VLAN Assignment, on page 597
- Configuration Examples for IEEE 802.1X VLAN Assignment, on page 601
- Additional References for IEEE 802.1X Port-Based Authentication, on page 602
- Feature History for IEEE 802.1X VLAN Assignment, on page 602


## Prerequisites for IEEE 802.1X VLAN Assignment

The following tasks must be completed before implementing the IEEE 802.1X VLAN Assignment feature:

- IEEE 802.1X must be enabled on the device port.
- The device must have a RADIUS configuration and be connected to the Cisco secure access control server (ACS). You should understand the concepts of the RADIUS protocol and have an understanding of how to create and apply access control lists (ACLs).
- EAP support must be enabled on the RADIUS server.
- You must configure the IEEE 802.1X supplicant to send an EAP-logoff (Stop) message to the switch when the user logs off. If you do not configure the IEEE 802.1X supplicant, an EAP-logoff message is not sent to the switch and the accompanying accounting Stop message is not sent to the authentication server. See the Microsoft Knowledge Base article at the location http://support.microsoft.com and set the SupplicantMode registry to 3 and the AuthMode registry to 1.
- Authentication, authorization, and accounting (AAA) must be configured on the port for all network-related service requests. The authentication method list must be enabled and specified. A method list describes the sequence and authentication method to be queried to authenticate a user. See the IEEE 802.1X Authenticator feature module for information.
- The port must be successfully authenticated.

The IEEE 802.1X VLAN Assignment feature is available only on Cisco 89x and 88x series integrated switching routers (ISRs) that support switch ports.

The following ISR-G2 routers are supported:

- 1900
- 2900
- 3900
- 3900e

The following cards or modules support switch ports:

- Enhanced High-speed WAN interface cards (EHWICs) with ACL support:
  - EHWIC-4ESG-P
  - EHWIC-9ESG-P
  - EHWIC-4ESG
  - EHWIC-9ESG
- High-speed WAN interface cards (HWICs) without ACL support:
  - HWIC-4ESW-P
  - HWIC-9ESW-P
  - HWIC-4ESW
  - HWIC-9ES

# Restrictions for IEEE 802.1X VLAN Assignment

- The IEEE 802.1X VLAN Assignment feature is available only on a switch port.
- The device port is always assigned to the configured access VLAN when any of the following conditions occurs:
  - No VLAN is supplied by the RADIUS server.
  - The VLAN information from the RADIUS server is not valid.
  - IEEE 802.1X authentication is disabled on the port.
  - The port is in the force authorized, force unauthorized, unauthorized, or shutdown state.

**Note** An access VLAN is a VLAN assigned to an access port. All packets sent from or received on this port belong to this VLAN.

- Assignment to the configured access VLAN prevents ports from appearing unexpectedly in an inappropriate VLAN because of a configuration error. Examples of configuration errors include the following:
  - A nonexistent or malformed VLAN ID
  - Attempted assignment to a voice VLAN ID
- When IEEE 802.1X authentication is enabled on a port, you cannot configure a port VLAN that is equal to a voice VLAN.
- If the multihost mode is enabled on an IEEE 802.1X port, all hosts are placed in the same VLAN (specified by the RADIUS server) as the first authenticated host.
- If an IEEE 802.1X port is authenticated and put in the RADIUS server-assigned VLAN, any change to the port access VLAN configuration does not take effect.
- This feature does not support standard ACLs on the switch port.

# Information About IEEE 802.1X VLAN Assignment

## Configuring Authorization

The AAA authorization feature is used to determine what a user can and cannot do. When AAA authorization is enabled, the network access server uses information retrieved from the user profile that is located either in the local user database or on the security server, to configure the user’s session. The user is granted access to a requested service only if the information in the user profile allows it.

## IEEE 802.1X Authentication with VLAN Assignment

Device ports support IEEE 802.1X authentication with VLAN assignment. After successful IEEE 802.1X authentication of a port, the RADIUS server sends the VLAN assignment to configure the device port.

The RADIUS server database maintains the username-to-VLAN mappings, assigning the VLAN based on the username of the supplicant connected to the device port.

# How to Configure IEEE 802.1X VLAN Assignment

## Enabling AAA Authorization for VLAN Assignment

AAA authorization limits the services available to a user. When AAA authorization is enabled, the device uses information retrieved from the user's profile, which is in the local user database or on the security server, to configure the user's session. The user is granted access to a requested service only if the information in the user profile allows it.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authorization network radius if-authenticated**<br>**Example:**<br>`Device(config)# aaa authorization network radius if-authenticated` | Configures the device for user RADIUS authorization for all network-related service requests. RADIUS authorization succeeds if the user has authenticated. |
| **Step 5** | **aaa authorization exec radius if-authenticated**<br>**Example:**<br>`Device(config)# aaa authorization exec radius if-authenticated` | Configures the device for user RADIUS authorization if the user has privileged EXEC access. RADIUS authorization succeeds if the user has authenticated. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Enabling IEEE 802.1X Authentication and Authorization

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **aaa authentication dot1x** {**default** \| *listname*} *method1* [*method2...*]<br>**Example:**<br>`Device(config)# aaa authentication dot1x default group radius` | Creates a series of authentication methods that are used to determine user privilege to access the privileged command level so that the device can communicate with the AAA server. |
| **Step 5** | **dot1x system-auth-control**<br>**Example:**<br>`Device(config)# dot1x system-auth-control` | Globally enables 802.1X port-based authentication. |
| **Step 6** | **identity profile default**<br>**Example:**<br>`Device(config)# identity profile default` | Creates an identity profile and enters dot1x profile configuration mode. |
| **Step 7** | **exit**<br>**Example:**<br>`Device(config-identity-prof)# exit` | Exits dot1x profile configuration mode and returns to global configuration mode. |
| **Step 8** | **interface** *type slot/port*<br>**Example:**<br>`Device(config)# interface Gigabitethernet 1/0/1` | Enters interface configuration mode and specifies the interface to be enabled for 802.1X authentication. |
| **Step 9** | **access-session port-control** {**auto** \| **force-authorized** \| **force-unauthorized**}<br>**Example:**<br>`Device(config-if)# access-session port-control auto` | Enables 802.1X port-based authentication on the interface.<br>• **auto**—Enables IEEE 802.1X authentication and causes the port to begin in the unauthorized state, allowing only EAPOL frames to be sent and received through the port. The authentication process begins when the link state of the port changes from down to up or when an EAPOL-start frame is received. The Device requests the identity of the supplicant and begins relaying authentication messages between the supplicant and the authentication server. Each supplicant attempting to access the |

| | Command or Action | Purpose |
|---|---|---|
| | | network is uniquely identified by the Device by using the supplicant MAC address.<br>• **force-authorized**-—Disables IEEE 802.1X authentication and causes the port to change to the authorized state without any authentication exchange required. The port sends and receives normal traffic without IEEE 802.1X-based authentication of the client. This is the default setting.<br>• **force-unauthorized**—Causes the port to remain in the unauthorized state, ignoring all attempts by the supplicant to authenticate. The Device cannot provide authentication services to the supplicant through the port. |
| **Step 10** | **dot1x pae** [**supplicant** \| **authenticator** \| **both**]<br>**Example:**<br>`Device(config-if)# dot1x pae authenticator` | Sets the Port Access Entity (PAE) type.<br>• **supplicant**—The interface acts only as a supplicant and does not respond to messages that are meant for an authenticator.<br>• **authenticator**-—The interface acts only as an authenticator and does not respond to any messages meant for a supplicant.<br>• **both**—The interface behaves both as a supplicant and as an authenticator and thus does respond to all dot1x messages. |
| **Step 11** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and enters privileged EXEC mode. |
| **Step 12** | **show dot1x**<br>**Example:**<br>`Device# show dot1x` | Displays whether 802.1X authentication has been configured on the device. |

# Specifying an Authorized VLAN in the RADIUS Server Database

The Internet Engineering Task Force (IETF) draft standard specifies a method for communicating vendor-specific information between the device and the RADIUS server by using the vendor-specific attribute (attribute 26). Vendor-specific attributes (VSAs) allow vendors to support their own extended attributes not

suitable for general use. The Cisco RADIUS implementation supports one vendor-specific option by using the format recommended in the specification.

- You must assign the following vendor-specific tunnel attributes in the RADIUS server database. The RADIUS server must return these attributes to the device:
  - [64] Tunnel-Type = VLAN
  - [65] Tunnel-Medium-Type = 802
  - [81] Tunnel-Private-Group-ID = VLAN name or VLAN ID

Attribute [64] must contain the value “VLAN” (type 13). Attribute [65] must contain the value “802” (type 6). Attribute [81] specifies the VLAN name or VLAN ID assigned to the IEEE 802.1X-authenticated user.

# Configuration Examples for IEEE 802.1X VLAN Assignment

## Example: Enabling AAA Authorization for VLAN Assignment

The following example shows how to enable AAA Authorization for VLAN assignment:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authorization network radius if-authenticated
Device(config)# aaa authorization exec radius if-authenticated
Device(config)# end
```

## Example: Enabling 802.1X Authentication

The following example shows how to enable 802.1X authentication on a device:

```
Device# configure terminal
Device(config)# aaa new-model
Device(config)# aaa authentication dot1x default group radius group radius
Device(config)# dot1x system-auth-control
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# dot1x port-control auto
```

The following **show dot1x** command output shows that 802.1X authentication has been configured on a device:

```
Device# show dot1x all

Sysauthcontrol              Enabled
Dot1x Protocol Version            2
Dot1x Info for GigabitEthernet 1/0/1
-----------------------------------
PAE                       = AUTHENTICATOR
PortControl               = AUTO
ControlDirection          = Both
HostMode                  = MULTI_HOST
```

```
ReAuthentication          = Enabled
QuietPeriod               = 600
ServerTimeout             = 60
SuppTimeout               = 30
ReAuthPeriod              = 1800 (Locally configured)
ReAuthMax                 = 2
MaxReq                    = 3
TxPeriod                  = 60
RateLimitPeriod           = 60
```

# Additional References for IEEE 802.1X Port-Based Authentication

### Standards and RFCs

| Standard/RFC | Title |
|---|---|
| IEEE 802.1X | *Port Based Network Access Control* |
| RFC 3580 | *IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines* |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br><br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br><br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/cisco/web/support/index.html |

# Feature History for IEEE 802.1X VLAN Assignment

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | IEEE 802.1X VLAN Assignment | The IEEE 802.1X VLAN Assignment feature is automatically enabled when IEEE 802.1X authentication is configured for an access port, which allows the RADIUS server to send a VLAN assignment to the device port. This assignment configures the device port so that network access can be limited for certain users.<br><br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | IEEE 802.1X VLAN Assignment | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 30

# Web-Based Authentication

This chapter describes how to configure web-based authentication on the device. It contains these sections:


- Restrictions for Web-Based Authentication, on page 605
- Information About Web-Based Authentication, on page 605
- How to Configure Web-Based Authentication, on page 614
- Verifying Web-Based Authentication, on page 626
- Feature History for Web-Based Authentication, on page 626


# Restrictions for Web-Based Authentication

A device without host switch virtual interface (SVI) does not intercept TCP SYN packets for Cisco Identity Services Engine (ISE) posture redirection.

# Information About Web-Based Authentication

## Web-Based Authentication Overview

Use the web-based authentication feature, known as web authentication proxy, to authenticate end users on host systems that do not run the IEEE 802.1x supplicant.

When you initiate an HTTP session, web-based authentication intercepts ingress HTTP packets from the host and sends an HTML login page to the users. The users enter their credentials, which the web-based authentication feature sends to the authentication, authorization, and accounting (AAA) server for authentication.

If authentication succeeds, web-based authentication sends a Login-Successful HTML page to the host and applies the access policies returned by the AAA server.

If authentication fails, web-based authentication forwards a Login-Fail HTML page to the user, prompting the user to retry the login. If the user exceeds the maximum number of attempts, web-based authentication forwards a Login-Expired HTML page to the host, and the user is placed on a watch list for a waiting period.

**Note** HTTPS traffic interception for central web authentication redirect is not supported.

**Note** You should use global parameter-map (for method-type, custom, and redirect) only for using the same web authentication methods like consent, web consent, and webauth, for all the clients and SSIDs. This ensures that all the clients have the same web-authentication method.

If the requirement is to use Consent for one SSID and Web-authentication for another SSID, then you should use two named parameter-maps. You should configure Consent in first parameter-map and configure webauth in second parameter-map.

**Note** The traceback that you receive when webauth client tries to do authentication does not have any performance or behavioral impact. It happens rarely when the context for which FFM replied back to EPM for ACL application is already dequeued (possibly due to timer expiry) and the session becomes 'unauthorized'.

Based on where the web pages are hosted, the local web authention can be categorozied as follows:

- *Internal*—The internal default HTML pages (Login, Success, Fail, and Expire) in the controller are used during the local web authentication.
- *Customized*—The customized web pages (Login, Success, Fail, and Expire) are downloaded onto the controller and used during the local web authentication.
- *External*—The customized web pages are hosted on the external web server instead of using the in-built or custom web pages.

Based on the various web authentication pages, the types of web authentication are as follows:

- *Webauth*—This is a basic web authentication. Herein, the controller presents a policy page with the user name and password. You need to enter the correct credentials to access the network.
- *Consent or web-passthrough*—Herein, the controller presents a policy page with the Accept or Deny buttons. You need to click the Accept button to access the network.
- *Webconsent*—This is a combination of webauth and consent web authentication types. Herein, the controller presents a policy page with Accept or Deny buttons along with user name or password. You need to enter the correct credentials and click the Accept button to access the network.

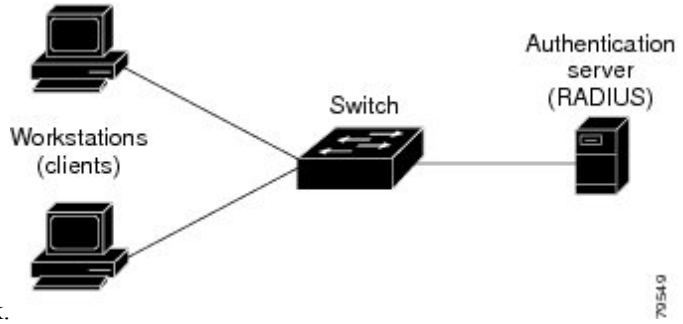
## Device Roles

With web-based authentication, the devices in the network have these specific roles:

- *Client*—The device (workstation) that requests access to the LAN and the services and responds to requests from the switch. The workstation must be running an HTML browser with Java Script enabled.
- *Authentication server*—Authenticates the client. The authentication server validates the identity of the client and notifies the switch that the client is authorized to access the LAN and the switch services or that the client is denied.
- *Switch*—Controls the physical access to the network based on the authentication status of the client. The switch acts as an intermediary (proxy) between the client and the authentication server, requesting identity information from the client, verifying that information with the authentication server, and relaying a response to the client.

*Figure 39: Web-Based Authentication Device Roles*

This figure shows the roles of these devices in a




network.

## Host Detection

The switch maintains an IP device tracking table to store information about detected hosts.

For Layer 2 interfaces, web-based authentication detects IP hosts by using these mechanisms:

- ARP based trigger—ARP redirect ACL allows web-based authentication to detect hosts with a static IP address or a dynamic IP address.
- Dynamic ARP inspection
- DHCP snooping—Web-based authentication is notified when the switch creates a DHCP-binding entry for the host.

## Session Creation

When web-based authentication detects a new host, it creates a session as follows:

- Reviews the exception list.

  If the host IP is included in the exception list, the policy from the exception list entry is applied, and the session is established.

- Reviews for authorization bypass

  If the host IP is not on the exception list, web-based authentication sends a nonresponsive-host (NRH) request to the server.

  If the server response is access accepted, authorization is bypassed for this host. The session is established.

- Sets up the HTTP intercept ACL

  If the server response to the NRH request is access rejected, the HTTP intercept ACL is activated, and the session waits for HTTP traffic from the host.

## Authentication Process

When you enable web-based authentication, these events occur:

- The user initiates an HTTP session.

- The HTTP traffic is intercepted, and authorization is initiated. The switch sends the login page to the user. The user enters a username and password, and the switch sends the entries to the authentication server.
- If the authentication succeeds, the switch downloads and activates the user's access policy from the authentication server. The login success page is sent to the user.
- If the authentication fails, the switch sends the login fail page. The user retries the login. If the maximum number of attempts fails, the switch sends the login expired page, and the host is placed in a watch list. After the watch list times out, the user can retry the authentication process.
- If the authentication server does not respond to the switch, and if an AAA fail policy is configured, the switch applies the failure access policy to the host. The login success page is sent to the user.
- The switch reauthenticates a client when the host does not respond to an ARP probe on a Layer 2 interface, or when the host does not send any traffic within the idle timeout on a Layer 3 interface.
- The switch reauthenticates a client when the host does not respond to an ARP probe on a Layer 2 interface.
- The feature applies the downloaded timeout or the locally configured session timeout.
- If the terminate action is RADIUS, the feature sends a nonresponsive host (NRH) request to the server. The terminate action is included in the response from the server.
- If the terminate action is default, the session is dismantled, and the applied policy is removed.

## Local Web Authentication Banner

With Web Authentication, you can create a default and customized web-browser banners that appears when you log in to a switch.

The banner appears on both the login page and the authentication-result pop-up pages. The default banner messages are as follows:

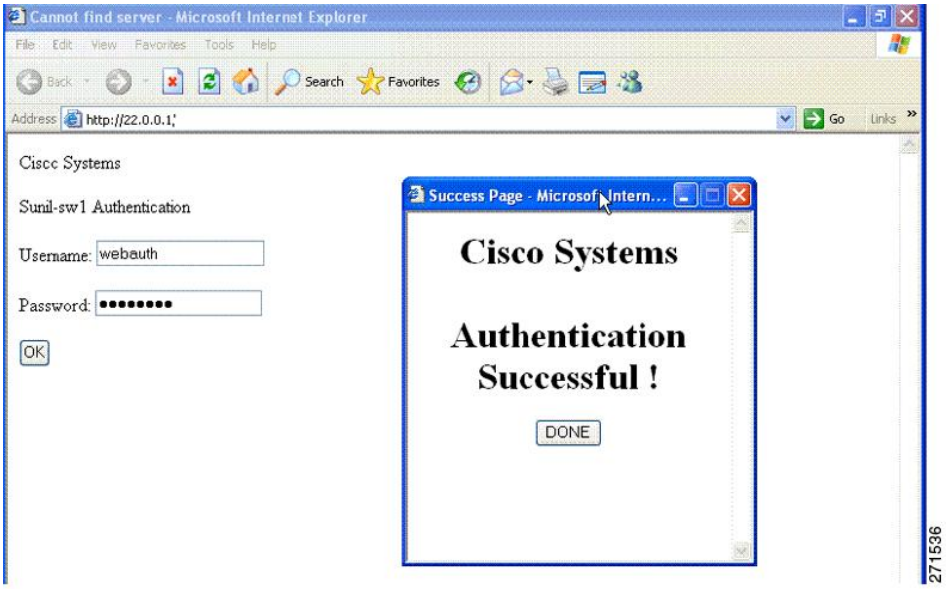
- *Authentication Successful*
- *Authentication Failed*
- *Authentication Expired*

The Local Web Authentication Banner can be configured in as follows:

- Legacy mode—Use the **ip admission auth-proxy-banner http** global configuration command.
- New-style mode—Use the **parameter-map type webauth global banner** global configuration command.

The default banner *Cisco Systems* and *Switch host-name Authentication* appear on the Login Page. *Cisco Systems* appears on the authentication result pop-up page.

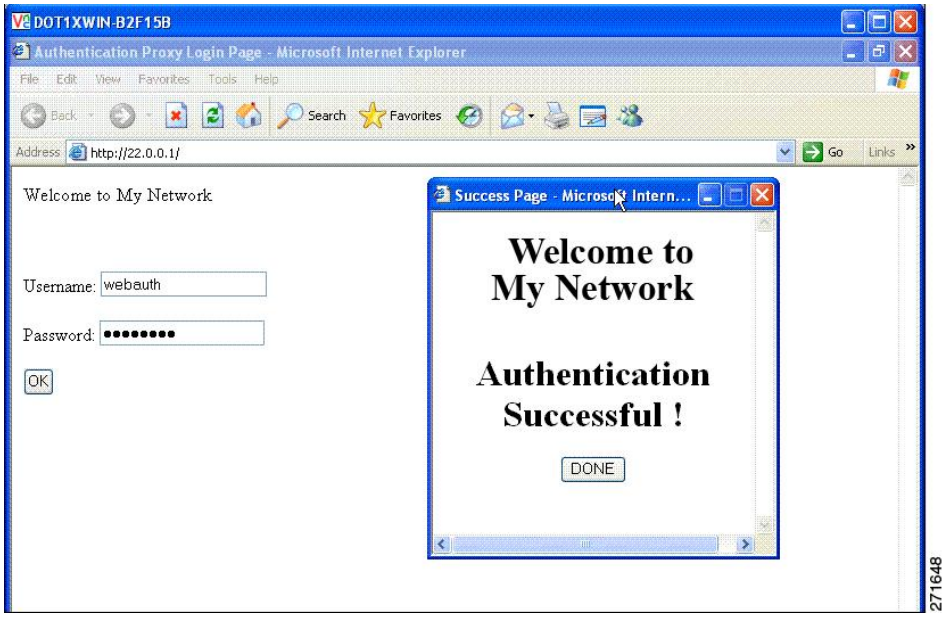
*Figure 40: Authentication Successful Banner*




The banner can be customized as follows:

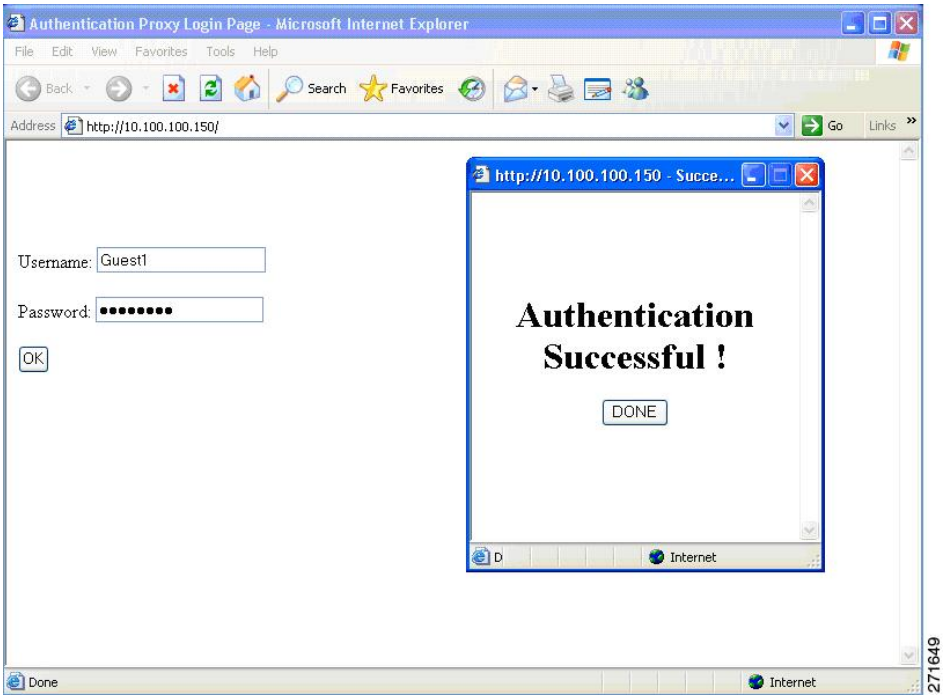
- Add a message, such as switch, router, or company name to the banner:
  - Legacy mode—Use the **ip admission auth-proxy-banner http** *banner-text*global configuration command.
  - New-style mode—Use the **parameter-map type webauth global banner** global configuration command.
- Add a logo or text file to the banner:
  - Legacy mode—Use the **ip admission auth-proxy-banner http** *file-path* global configuration command.
  - New-style mode—Use the **parameter-map type webauth global banner** global configuration command.

*Figure 41: Customized Web Banner*




If you do not enable a banner, only the username and password dialog boxes appear in the web authentication login screen, and no banner appears when you log into the switch.

*Figure 42: Login Screen With No Banner*



# Web Authentication Customizable Web Pages

During the web-based authentication process, the switch internal HTTP server hosts four HTML pages to deliver to an authenticating client. The server uses these pages to notify you of these four-authentication process states:

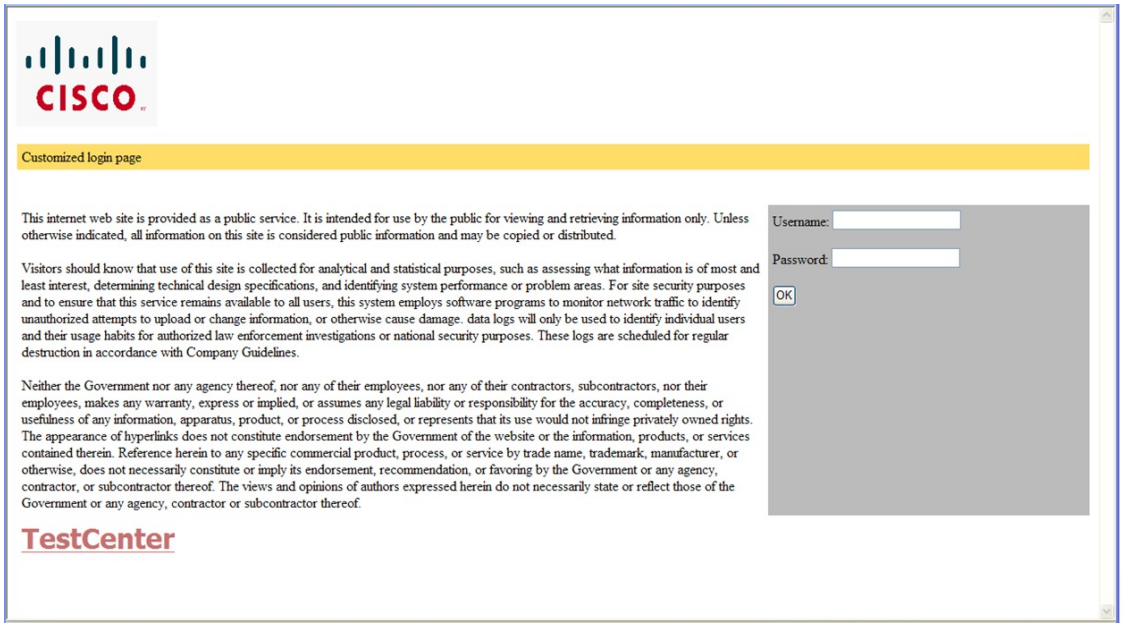
- Login—Your credentials are requested.
- Success—The login was successful.
- Fail—The login failed.
- Expire—The login session has expired because of excessive login failures.

## Guidelines

- You can substitute your own HTML pages for the default internal HTML pages.
- You can use a logo or specify text in the *login*, *success*, *failure*, and *expire* web pages.
- On the banner page, you can specify text in the login page.
- The pages are in HTML.
- You must include an HTML redirect command in the success page to access a specific URL.
- The URL string must be a valid URL (for example, http://www.cisco.com). An incomplete URL might cause *page not found* or similar errors on a web browser.
- If you configure web pages for HTTP authentication, they must include the appropriate HTML commands (for example, to set the page time out, to set a hidden password, or to confirm that the same page is not submitted twice).
- The CLI command to redirect users to a specific URL is not available when the configured login form is enabled. The administrator should ensure that the redirection is configured in the web page.
- If the CLI command redirecting users to specific URL after authentication occurs is entered and then the command configuring web pages is entered, the CLI command redirecting users to a specific URL does not take effect.
- Configured web pages can be copied to the switch boot flash or flash.
- The login page can be on one flash, and the success and failure pages can be another flash (for example, the flash on the active switch or a member switch).
- You must configure all four pages.
- The banner page has no effect if it is configured with the web page.
- All of the logo files (image, flash, audio, video, and so on) that are stored in the system directory (for example, flash, disk0, or disk) and that must be displayed on the login page must use *web_auth_<filename>* as the file name.
- The configured authentication proxy feature supports both HTTP and SSL.

You can substitute your HTML pages for the default internal HTML pages. You can also specify a URL to which users are redirected after authentication occurs, which replaces the internal Success page.

*Figure 43: Customizable Authentication Page*




## Authentication Proxy Web Page Guidelines

When configuring customized authentication proxy web pages, follow these guidelines:

- To enable the custom web pages feature, specify all four custom HTML files. If you specify fewer than four files, the internal default HTML pages are used.
- The four custom HTML files must be present on the flash memory of the switch. The maximum size of each HTML file is 8 KB.
- Any images on the custom pages must be on an accessible HTTP server. Configure an intercept ACL within the admission rule.
- Any external link from a custom page requires configuration of an intercept ACL within the admission rule.
- To access a valid DNS server, any name resolution required for external links or images requires configuration of an intercept ACL within the admission rule.
- If the custom web pages feature is enabled, a configured auth-proxy-banner is not used.
- If the custom web pages feature is enabled, the redirection URL for successful login feature is not available.
- To remove the specification of a custom file, use the **no** form of the command.

Because the custom login page is a public web form, consider these guidelines for the page:

- The login form must accept user entries for the username and password and must show them as **uname** and **pwd**.
- The custom login page should follow best practices for a web form, such as page timeout, hidden password, and prevention of redundant submissions.

## Redirection URL for Successful Login Guidelines

When configuring a redirection URL for successful login, consider these guidelines:

- If the custom authentication proxy web pages feature is enabled, the redirection URL feature is disabled and is not available in the CLI. You can perform redirection in the custom-login success page.
- If the redirection URL feature is enabled, a configured auth-proxy-banner is not used
- To remove the specification of a redirection URL, use the **no** form of the command.
- If the redirection URL is required after the web-based authentication client is successfully authenticated, then the URL string must start with a valid URL (for example, http://) followed by the URL information. If only the URL is given without http://, then the redirection URL on successful authentication might cause page not found or similar errors on a web browser.

# Web-based Authentication Interactions with Other Features

## Port Security

You can configure web-based authentication and port security on the same port. Web-based authentication authenticates the port, and port security manages network access for all MAC addresses, including that of the client. You can then limit the number or group of clients that can access the network through the port.

## LAN Port IP

You can configure LAN port IP (LPIP) and Layer 2 web-based authentication on the same port. The host is authenticated by using web-based authentication first, followed by LPIP posture validation. The LPIP host policy overrides the web-based authentication host policy.

If the web-based authentication idle timer expires, the NAC policy is removed. The host is authenticated, and posture is validated again.

## Gateway IP

You cannot configure Gateway IP (GWIP) on a Layer 3 VLAN interface if web-based authentication is configured on any of the switch ports in the VLAN.

You can configure web-based authentication on the same Layer 3 interface as Gateway IP. The host policies for both features are applied in software. The GWIP policy overrides the web-based authentication host policy.

## ACLs

If you configure a VLAN ACL or a Cisco IOS ACL on an interface, the ACL is applied to the host traffic only after the web-based authentication host policy is applied.

For Layer 2 web-based authentication, it is more secure, though not required, to configure a port ACL (PACL) as the default access policy for ingress traffic from hosts connected to the port. After authentication, the web-based authentication host policy overrides the PACL. The Policy ACL is applied to the session even if there is no ACL configured on the port.

You cannot configure a MAC ACL and web-based authentication on the same interface.

You cannot configure web-based authentication on a port whose access VLAN is configured for VACL capture.

## EtherChannel

You can configure web-based authentication on a Layer 2 EtherChannel interface. The web-based authentication configuration applies to all member channels.

# How to Configure Web-Based Authentication

## Default Web-Based Authentication Configuration

The following table shows the default web-based authentication configuration.

***Table 31: Default Web-based Authentication Configuration***

| Feature | Default Setting |
|---|---|
| AAA | Disabled |
| RADIUS server<br>• IP address<br>• UDP authentication port<br>• Key | • None specified<br>• None specified |
| Default value of inactivity timeout | 3600 seconds |
| Inactivity timeout | Enabled |

## Web-Based Authentication Configuration Guidelines and Restrictions

- Web-based authentication is an ingress-only feature.
- You can configure web-based authentication only on access ports. Web-based authentication is not supported on trunk ports, EtherChannel member ports, or dynamic trunk ports.
- External web authentication, where the switch redirects a client to a particular host or web server for displaying login message, is not supported.
- You cannot authenticate hosts on Layer 2 interfaces with static ARP cache assignment. These hosts are not detected by the web-based authentication feature because they do not send ARP messages.
- By default, the IP device tracking feature is disabled on a switch. You must enable the IP device tracking feature to use web-based authentication.
- You must enable SISF-Based device tracking to use web-based authentication. By default, SISF-Based device tracking is disabled on a switch.
- You must configure at least one IP address to run the switch HTTP server. You must also configure routes to reach each host IP address. The HTTP server sends the HTTP login page to the host.

- Hosts that are more than one hop away might experience traffic disruption if an STP topology change results in the host traffic arriving on a different port. This occurs because the ARP and DHCP updates might not be sent after a Layer 2 (STP) topology change.
- Web-based authentication does not support VLAN assignment as a downloadable-host policy.
- Web-based authentication supports IPv6 in Session-aware policy mode. IPv6 Web-authentication requires at least one IPv6 address configured on the switch and IPv6 Snooping configured on the switchport.
- Web-based authentication and Network Edge Access Topology (NEAT) are mutually exclusive. You cannot use web-based authentication when NEAT is enabled on an interface, and you cannot use NEAT when web-based authentication is running on an interface.
- Identify the following RADIUS security server settings that will be used while configuring switch-to-RADIUS-server communication:
  - Host name
  - Host IP address
  - Host name and specific UDP port numbers
  - IP address and specific UDP port numbers

  The combination of the IP address and UDP port number creates a unique identifier, that enables RADIUS requests to be sent to multiple UDP ports on a server at the same IP address. If two different host entries on the same RADIUS server are configured for the same service (for example, authentication) the second host entry that is configured functions as the failover backup to the first one. The RADIUS host entries are chosen in the order that they were configured.
- When you configure the RADIUS server parameters:
  - Specify the **key** *string* on a separate command line.
  - For **key** *string*, specify the authentication and encryption key used between the switch and the RADIUS daemon running on the RADIUS server. The key is a text string that must match the encryption key used on the RADIUS server.
  - When you specify the **key** *string*, use spaces within and at the end of the key. If you use spaces in the key, do not enclose the key in quotation marks unless the quotation marks are part of the key. This key must match the encryption used on the RADIUS daemon.
  - You can globally configure the timeout, retransmission, and encryption key values for all RADIUS servers by using with the **radius-server host** global configuration command. If you want to configure these options on a per-server basis, use the **radius-server timeout**, radius-server transmit, and the radius-server key global configuration commands.

**Note** You need to configure some settings on the RADIUS server, including: the switch IP address, the key string to be shared by both the server and the switch, and the downloadable ACL (DACL). For more information, see the RADIUS server documentation.

- For a URL redirect ACL:

- Packets that match a permit access control entry (ACE) rule are sent to the CPU for forwarding to the AAA server.
- Packets that match a deny ACE rule are forwarded through the switch.
- Packets that match neither the permit ACE rule or deny ACE rule are processed by the next dACL, and if there is no dACL, the packets hit the implicit-deny ACL and are dropped.

# Configuring the Authentication Rule and Interfaces

Follow these steps to configure the authentication rule and interfaces:

### Before you begin

SISF-Based device tracking is a prerequisite to Web Authentication. Ensure that you have enabled device tracking programmatically or manually.

For more information, see *Confguring SISF-Based Tracking*.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip admission name** *name* **proxy http**<br>**Example:**<br>`Device(config)# ip admission name webauth1 proxy http` | Configures an authentication rule for web-based authorization. |
| **Step 4** | **interface** *type slot/port*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Enters interface configuration mode and specifies the ingress Layer 2 or Layer 3 interface to be enabled for web-based authentication.<br>*type* can be FastEthernet, GigabitEthernet, or TenGigabitEthernet. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **ip access-group** *name*<br>**Example:**<br>`Device(config-if)# ip access-group webauthag` | Applies the default ACL. |
| **Step 6** | **ip admission name**<br>**Example:**<br>`Device(config)# ip admission name` | Configures an authentication rule for web-based authorization for the interface. |
| **Step 7** | **exit**<br>**Example:**<br>`Device# exit` | Exits global configuration mode and returns to privileged EXEC mode. |
| **Step 8** | **show ip admission**<br>**Example:**<br>`Device# show ip admission` | Displays the network admission cache entries and information about web authentication sessions. |

# Configuring AAA Authentication

If a method-list is configured under VTY lines, the corresponding method list must be added to the AAA configuration:

```
Device(config)# line vty 0 4
Device(config-line)# authorization commands 15 list1
Device(config-line)# exit
Device(config)# aaa authorization commands 15 list1 group tacacs+
```

If a method-list is not configured under VTY lines, you must add the default method list to the AAA configuration:

```
Device(config)# line vty 0 4
Device(config-line)# exit
Device(config)# aaa authorization commands 15 default group tacacs+
```

Follow these steps to configure AAA authentication:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** | Enables privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device> enable` | • Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA functionality. |
| **Step 4** | **aaa authentication login default group** {**tacacs+** \| **radius**}<br>**Example:**<br>`Device(config)# aaa authentication login default group tacacs+` | Defines the list of authentication methods at login.<br>**named_authentication_list** refers to any name that is not greater than 31 characters.<br>**AAA_group_name** refers to the server group name. You need to define the server-group **server_name** at the beginning itself. |
| **Step 5** | **aaa authorization auth-proxy default group** {**tacacs+** \| **radius**}<br>**Example:**<br>`Device(config)# aaa authorization auth-proxy default group tacacs+` | Creates an authorization method list for web-based authorization. |
| **Step 6** | **tacacs server** *server-name*<br>**Example:**<br>`Device(config)# tacacs server yourserver` | Specifies an AAA server. |
| **Step 7** | **address** {**ipv4 \| ipv6**} *ip address*<br>**Example:**<br>`Device(config-server-tacacs)# address ipv4 10.0.1.12` | Configures the IP address for the TACACS server. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **key** *string*<br>**Example:**<br>`Device(config-server-tacacs)# key cisco123` | Configures the authorization and encryption key used between the switch and the TACACS server. |
| **Step 9** | **end**<br>**Example:**<br>`Device(config-server-tacacs)# end` | Exits the TACACS server mode and returns to privileged EXEC mode. |

# Configuring Switch-to-RADIUS-Server Communication

Follow these steps to configure the RADIUS server parameters:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip radius source-interface vlan** *vlan interface number*<br>**Example:**<br>`Device(config)# ip radius source-interface vlan 80` | Specifies that the RADIUS packets have the IP address of the indicated interface. |
| **Step 4** | **radius server** *server name*<br>**Example:**<br>`Device(config)# radius server rsim address ipv4 124.2.2.12` | (Optional) Specifies the IP address of the RADIUS server. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **address** {**ipv4** \| **ipv6**} *ip address*<br>**Example:**<br>`Device(config-radius-server)# address ipv4 10.0.1.2 auth-port 1550 acct-port 1560` | Configures the IP address for the RADIUS server. |
| **Step 6** | **key** *string*<br>**Example:**<br>`Device(config-radius-server)# key rad123` | (Optional) Specifies the authentication and encryption key used between the switch and the RADIUS daemon running on the RADIUS server. |
| **Step 7** | **exit**<br>**Example:**<br>`Device(config-radius-server)# exit` | Exits the RADIUS server mode and enters the global configuration mode. |
| **Step 8** | **radius-server vsa send authentication** *string*<br>**Example:**<br>`Device(config)# radius-server vsa send authentication` | Enable downloading of an ACL from the RADIUS server. |
| **Step 9** | **radius-server dead-criteria** [**time** *seconds*] [**tries** *num-tries*]<br>**Example:**<br>`Device(config)# radius-server dead-criteria tries 45` | Configures the conditions that determine when a RADIUS server is considered unavailable or dead.<br>Enter **time** in seconds during which there is no response from RADIUS server to the device.<br>Enter number of **tries** where there will be no valid response from RADIUS server to the device. The range of *num-tries* is 1 to 100.<br>**Note**<br>The device will shut down if the total number of tries is set to 45 or lower when the device is part of a stack. We recommend you to enter a longer duration of time. Higher value of number of tries prevents the device from shutting down while booting. |
| **Step 10** | **end**<br>**Example:** | Exits global configuration mode and returns to privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# end` | |

# Configuring the HTTP Server

To use web-based authentication, you must enable the HTTP server within the device. You can enable the server for either HTTP or HTTPS.

**Note** The Apple psuedo-browser will not open if you configure only the **ip http secure-server** command. You should also configure the **ip http server** command.

Follow the procedure given below to enable the server for either HTTP or HTTPS:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip http server**<br>**Example:**<br>`Device(config)# ip http server` | Enables the HTTP server. The web-based authentication feature uses the HTTP server to communicate with the hosts for user authentication. |
| **Step 4** | **ip http secure-server**<br>**Example:**<br>`Device(config)# ip http secure-server` | Enables HTTPS.<br>You can configure custom authentication proxy web pages or specify a redirection URL for successful login.<br>**Note**<br>To ensure secure authentication when you enter the **ip http secure-server** command, the login page is always in HTTPS (secure HTTP) even if the user sends an HTTP request. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br>**Example:**<br>`Device# end` | Exits global configuration mode and returns to privileged EXEC mode. |

## Customizing the Authentication Proxy Web Pages

You can configure web authentication to display four substitute HTML pages to the user in place of the default HTML pages during web-based authentication.

Follow these steps to specify the use of your custom authentication proxy web pages:

### Before you begin

Store your custom HTML files on the device flash memory.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip admission proxy http login page file** *device:login-filename*<br>**Example:**<br>`Device(config)# ip admission proxy http login page file disk1:login.htm` | Specifies the location in the device memory file system of the custom HTML file to use in place of the default login page. The *device:* is flash memory. |
| **Step 4** | **ip admission proxy http success page file** *device:success-filename*<br>**Example:**<br>`Device(config)# ip admission proxy http success page file disk1:success.htm` | Specifies the location of the custom HTML file to use in place of the default login success page. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **ip admission proxy http failure page file** *device:fail-filename*<br>**Example:**<br>`Device(config)# ip admission proxy http fail page file disk1:fail.htm` | Specifies the location of the custom HTML file to use in place of the default login failure page. |
| **Step 6** | **ip admission proxy http login expired page file** *device:expired-filename*<br>**Example:**<br>`Device(config)# ip admission proxy http login expired page file disk1:expired.htm` | Specifies the location of the custom HTML file to use in place of the default login expired page. |
| **Step 7** | **end**<br>**Example:**<br>`Device# end` | Exits global configuration mode and returns to privileged EXEC mode. |

## Specifying a Redirection URL for a Successful Login

Follow these steps to specify a URL to which the user is redirected after authentication, effectively replacing the internal Success HTML page:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip admission proxy http success redirect** *url-string*<br>**Example:** | Specifies a URL for redirection of the user in place of the default login success page. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# ip admission proxy http success redirect www.example.com` | |
| **Step 4** | **end**<br>**Example:**<br>`Device# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring Web-Based Authentication Parameters

Follow these steps to configure the maximum number of failed login attempts before the client is placed in a watch list for a waiting period:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip admission max-login-attempts** *number*<br>**Example:**<br>`Device(config)# ip admission max-login-attempts 10` | Sets the maximum number of failed login attempts. The range is 1 to 2147483647 attempts. The default is 5. |
| **Step 4** | **exit**<br>**Example:**<br>`Device# exit` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring a Web-Based Authentication Local Banner

Follow these steps to configure a local banner on a switch that has web authentication configured.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ip admission auth-proxy-banner http** [*banner-text* \| *file-path*]<br>**Example:**<br>`Device(config)# ip admission auth-proxy-banner http C My Switch C` | Enables the local banner.<br>(Optional) Create a custom banner by entering *C banner-text C* (where *C* is a delimiting character), or *file-path* that indicates a file (for example, a logo or text file) that appears in the banner. |
| **Step 4** | **end**<br>**Example:**<br>`Device# end` | Exits global configuration mode and returns to privileged EXEC mode. |

## Removing Web-Based Authentication Cache Entries

Follow these steps to remove web-based authentication cache entries:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **clear ip auth-proxy cache** {* \| *host ip address*}<br>**Example:**<br>`Device# clear ip auth-proxy cache` | Delete authentication proxy entries. Use an asterisk to delete all cache entries. Enter a specific IP address to delete the entry for a single host. |

| | Command or Action | Purpose |
|---|---|---|
| | `192.168.4.5` | |
| **Step 3** | **clear ip admission cache** {* \| *host ip address*}<br>**Example:**<br>`# clear ip admission cache 192.168.4.5` | Delete authentication proxy entries. Use an asterisk to delete all cache entries. Enter a specific IP address to delete the entry for a single host. |

# Verifying Web-Based Authentication

Use the commands in this topic to display the web-based authentication settings for all interfaces or for specific ports.

***Table 32: Privileged EXEC show Commands***

| Command | Purpose |
|---|---|
| **show authentication sessions method webauth** | Displays the web-based authentication settings for all interfaces for fastethernet, gigabitethernet, or tengigabitethernet |
| **show authentication sessions interface** *type slot/port*[**details**] | Displays the web-based authentication settings for the specified interface for fastethernet, gigabitethernet, or tengigabitethernet.<br>In Session Aware Networking mode, use the **show access-session interface** command. |

# Feature History for Web-Based Authentication

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Web-Based Authentication | The Web-Based Authentication feature authenticates end users on host systems that do not run the IEEE 802.1x supplicant.<br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Web-Based Authentication | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 31

# Identity Based Networking Services Overview

Cisco Identity Based Networking Services (IBNS) provides a policy and identity-based framework in which edge devices can deliver flexible and scalable services to subscribers. This module provides information about what Cisco IBNS is and its features and benefits.


- Information About Identity-Based Networking Services, on page 627
- Feature History for Cisco Identity Based Networking Services Overview, on page 629

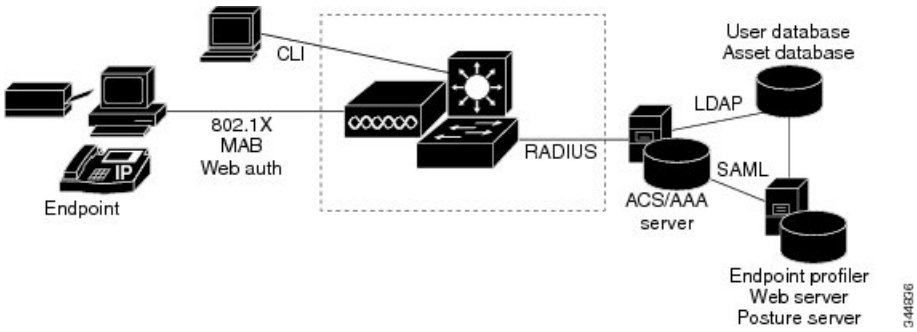

# Information About Identity-Based Networking Services

## Understanding Cisco Identity Based Networking Services

Cisco IBNS feature provides a policy and identity-based framework in which edge devices can deliver flexible and scalable services to subscribers. Cisco IBNS provides an identity-based approach to access management and subscriber management. It offers a consistent way to configure features across technologies, a command interface that allows easy deployment and customization of features, and a robust policy control engine with the ability to apply policies defined locally or received from an external server to enforce policy in the network.

The figure below illustrates a typical deployment of Cisco IBNS in a physically distributed enterprise with a campus, branch offices, and remote workers.

***Figure 44: Sample Deployment of Cisco IBNS***




## Features in Cisco Identity Based Networking Services

Cisco IBNS includes the following features:

- Cisco common classification policy language (C3PL)-based identity configuration
- Concurrent authentication methods on a single session, including IEEE 802.1x (dot1x), MAC authentication bypass (MAB), and web authentication
- Downloadable identity service templates
- Extended RADIUS change of authorization (CoA) support for querying, reauthenticating, and terminating a session, port shutdown and port bounce, and activating and deactivating an identity service template.
- Local authentication using Lightweight Directory Access Protocol (LDAP)
- Locally defined identity control policies
- Locally defined identity service templates
- Per-user inactivity handling across methods
- Web authentication support of common session ID
- Web authentication support of IPv6

# Benefits of Cisco Identity Based Networking Services

Identity-based solutions are essential for delivering access control for disparate groups such as employees, contractors, and partners while maintaining low operating expenses. Cisco IBNS provides a consistent approach to operational management through a policy and identity-based infrastructure leading to faster deployment of new features and easier management of switches.

Cisco IBNS provides the following benefits:

- An identity-based framework for session management.
- A robust policy control engine to apply policies defined locally or received from an external AAA server.
- Faster deployment and customization of features across access technologies.
- A simpler and consistent way to configure features across access methods, platforms, and application domains.

# Web Authentication Support for Common Session ID

Cisco IBNS allows a single session identifier to be used for web authentication sessions in addition to all 802.1X and MAB authenticated sessions for a client. This session ID is used for all reporting purposes such as show commands, MIBs, and RADIUS messages and allows users to distinguish messages for one session from messages for other sessions. This common session ID is used consistently across all authentication methods and features applied to a session.

# Web Authentication Support of IPv6

Cisco IBNS introduces IPv6 support for web authentication. IPv6 is supported for web authentication only when Cisco IBNS is explicitly configured. This means that you must permanently convert your configuration to the Cisco common classification policy language (C3PL) display mode by specifically configuring a Cisco IBNS command such as the **policy-map type control subscriber** command.

# IP Device Tracking

IP device tracking can be configured using the Switch Integrated Security Features (SISF) policy. Use the tracking enable command in device tracking configuration mode, to configure device tracking using SISF policy. Use the **show device-tracking** command to display the device tracking configuration.

The following is the sample configuration for device tracking.

```
Device(config)# device-tracking policy sisf_policy
Device(config-device-tracking)# tracking enable
Device(config-device-tracking)# exit
Device(config)# interface GigabitEthernet 3/0/1
Device(config-if)# switchport mode access
Device(config-if)# device-tracking attach-policy sisf_policy
Device(config-if)# end
```

# Feature History for Cisco Identity Based Networking Services Overview

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

***Table 33: Feature History for Cisco Identity Based Networking Services Overview***

| Release | Feature Name | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Web Authentication Support of Common Session ID | Allows a single session identifier to be used for all web authentication sessions in addition to 802.1X and MAB authenticated sessions.<br>Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches |
| Cisco IOS XE Fuji 16.8.1a | Web Authentication Support of Common Session ID | Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 32

# Change of Authorization Support

Authentication provides a method to identify users, which includes the login and password dialog, challenge and response, messaging support, and encryption, depending on the selected security protocol. Authentication is the way a user is identified prior to being allowed access to the network and network services.

Cisco Identity Based Networking Services (IBNS) supports RADIUS change of authorization (CoA) commands for session query, reauthentication, and termination, port bounce and port shutdown, and service template activation and deactivation. This module provides information about the supported CoA commands for Cisco IBNS.


- Information About CoA Support, on page 631
- Feature History for CoA Support, on page 635


# Information About CoA Support

## RADIUS Change-of-Authorization Support

Cisco IOS software supports the RADIUS CoA extensions defined in RFC 5176 that are typically used in a push model to allow the dynamic reconfiguring of sessions from external AAA or policy servers. Per-session CoA requests are supported for session identification, session termination, host reauthentication, port shutdown, and port bounce. This model comprises one request (CoA-Request) and two possible response codes:

- CoA acknowledgement (ACK) [CoA-ACK]
- CoA nonacknowledgement (NAK) [CoA-NAK]

The request is initiated from a CoA client (typically a AAA or policy server) and directed to the device that acts as a listener.

The table below shows the RADIUS CoA commands and vendor-specific attributes (VSAs) supported by Cisco IBNS. All CoA commands must include the session identifier between the device and the CoA client.

*Table 34: RADIUS CoA Commands Supported by Cisco IBNS*

| CoA Command | Cisco VSA |
|---|---|
| Activate service | Cisco:Avpair="subscriber:command=activate-service"<br>Cisco:Avpair="subscriber:service-name=<*service-name*>"<br>Cisco:Avpair="subscriber:precedence=<*precedence-number*>"<br>Cisco:Avpair="subscriber:activation-mode=replace-all" |
| Deactivate service | Cisco:Avpair="subscriber:command=deactivate-service"<br>Cisco:Avpair="subscriber:service-name=<*service-name*>" |
| Bounce host port | Cisco:Avpair="subscriber:command=bounce-host-port" |
| Disable host port | Cisco:Avpair="subscriber:command=disable-host-port" |
| Session query | Cisco:Avpair="subscriber:command=session-query" |
| Session reauthenticate | Cisco:Avpair="subscriber:command=reauthenticate"<br>Cisco:Avpair="subscriber:reauthenticate-type=last" or<br>Cisco:Avpair="subscriber:reauthenticate-type=rerun" |
| Session terminate | This is a standard disconnect request and does not require a VSA. |
| Interface template | Cisco:AVpair="interface-template-name=<*interfacetemplate*>" |

# Session Identification

For disconnect and CoA requests targeted at a particular session, the device locates the session based on one or more of the following attributes:

- Acct-Session-Id (IETF attribute #44)
- Audit-Session-Id (Cisco VSA)
- Calling-Station-Id (IETF attribute #31, which contains the host MAC address)
- IPv6 Attributes, which can be one of the following:
  - Framed-IPv6-Prefix (IETF attribute #97) and Framed-Interface-Id (IETF attribute #96), which together create a full IPv6 address per RFC 3162
  - Framed-IPv6-Address
- Plain IP Address (IETF attribute #8)

If more than one session identification attribute is included in the message, all of the attributes must match the session or the device returns a Disconnect-NAK or CoA-NAK with the error code Invalid Attribute Value.

For CoA requests targeted at a particular enforcement policy, the device returns a CoA-NAK with the error code Invalid Attribute Value if any of the above session identification attributes are included in the message.

# CoA Activate Service Command

The CoA activate service command can be used to activate a service template on a session. The AAA server sends the request in a standard CoA-Request message using the following VSAs:

Cisco:Avpair="subscriber:command=activate-service"

Cisco:Avpair="subscriber:service-name=<*service-name*>"

Cisco:Avpair="subscriber:precedence=<*precedence-number*>"

Cisco:Avpair="subscriber:activation-mode=replace-all"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the Session Identification, on page 632 section. If the device cannot locate a session, it returns a CoA-NAK message with the Session Context Not Found error-code attribute. If the device locates a session, it initiates an activate template operation for the hosting port and a CoA-ACK is returned. If activating the template fails, a CoA-NAK message is returned with the Error-Code attribute set to the appropriate message.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

# CoA Deactivate Service Command

The CoA deactivate service command can be used to deactivate a service template on a session. The AAA server sends the request in a standard CoA-Request message using the following VSAs:

Cisco:Avpair="subscriber:command=deactivate-service"

Cisco:Avpair="subscriber:service-name=<*service-name*>"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the Session Identification, on page 632 section. If the device cannot locate a session, it returns a CoA-NAK message with the Session Context Not Found error-code attribute. If the device locates a session, it initiates a deactivate template operation for the hosting port and a CoA-ACK is returned. If deactivating the template fails, a CoA-NAK message is returned with the Error-Code attribute set to the appropriate message.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

# CoA Bounce Host Port Command

The CoA bounce host port command terminates a session and bounces the port (initiates a link down event followed by a link up event). The AAA server sends the request in a standard CoA-Request message with the following VSA:

Cisco:Avpair="subscriber:command=bounce-host-port"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the Session Identification, on page 632 section. If the session cannot be located, the device returns a CoA-NAK message with the Session Context Not Found error-code attribute. If the session is located, the device disables the hosting port for a period of ten seconds, reenables it (port bounce), and returns a CoA-ACK.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

The CoA bounce port command is useful as a last resort when an endpoint needs to acquire a new IP address after a change in authorization and this is the only way to indicate to the endpoint to restart the DHCP process. This can occur when there is a VLAN change and the endpoint is a device, such as a printer, that does not have a mechanism to detect a change on this authentication port. This command can cause a link flap on an authentication port, which triggers DHCP renegotiation from one or more hosts connected to this port.

# CoA Disable Host Port Command

The CoA disable host port command administratively shuts down the authentication port that is hosting a session, which terminates the session. The AAA server sends the request in a standard CoA-Request message with the following VSA:

Cisco:Avpair="subscriber:command=disable-host-port"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the Session Identification, on page 632 section. If the device cannot locate the session, it returns a CoA-NAK message with the Session Context Not Found error-code attribute. If the device locates the session, it disables the hosting port and returns a CoA-ACK message.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

# CoA Session Query Command

The CoA session query command requests service information about a subscriber session. The AAA server sends the request in a standard CoA-Request message containing the following VSA:

Cisco:Avpair="subscriber:command=session-query"

Because this command is session-oriented, it must be accompanied by one or more of the session identification attributes described in the Session Identification, on page 632 section. If the device cannot locate a session, it returns a CoA-NAK message with the Session Context Not Found error-code attribute. If the device locates a session, it performs a session query operation on the session and returns a CoA-ACK message.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

# CoA Session Reauthenticate Command

To initiate session authentication, the AAA server sends a standard CoA-Request message containing the following VSAs:

Cisco:Avpair="subscriber:command=reauthenticate"

Cisco:Avpair="subscriber:reauthenticate-type=<last | rerun>"

"reauthenticate-type" defines whether the CoA reauthentication request uses the authentication method that last succeeded on the session or whether the authentication process is completely rerun.

The following rules apply:

- “subscriber:command=reauthenticate” must be present to trigger a reauthentication.
- If “subscriber:reauthenticate-type” is not specified, the default behavior is to rerun the last successful authentication method for the session. If the method reauthenticates successfully, all old authorization data is replaced with the new reauthenticated authorization data.
- “subscriber:reauthenticate-type” is valid only when included with “subscriber:command=reauthenticate.” If it is included in another CoA command, the VSA will be silently ignored.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is resent from the client. If the device fails after returning a CoA-ACK message to the client but before the operation is complete, the operation is restarted on the new active device.

## CoA Session Terminate Command

A CoA Disconnect-Request command terminates a session without disabling the host port. This command causes reinitialization of the authenticator state machine for the specified host, but does not restrict the host’s access to the network. If the session cannot be located, the device returns a Disconnect-NAK message with the Session Context Not Found error-code attribute. If the session is located, the device terminates the session. After the session has been completely removed, the device returns a Disconnect-ACK.

If the device fails before returning a CoA-ACK to the client, the process is repeated on the new active device when the request is re-sent from the client.

To restrict a host’s access to the network, use a CoA Request with the Cisco:Avpair=“subscriber:command=disable-host-port” VSA. This command is useful when a host is known to cause problems on the network and network access needs to be immediately blocked for the host. When you want to restore network access on the port, reenable it using a non-RADIUS mechanism.

# Feature History for CoA Support

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

*Table 35: Feature History for CoA Support*

| Release | Feature Name | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Change of Authorization | Supports CoA requests for initiating the following:<br>• Activating and deactivating service templates on sessions<br>• Port bounce<br>• Port shutdown<br>• Querying a session<br>• Reauthenticating a session<br>• Terminating a session<br>These VSAs are sent in a standard CoA-Request message from a AAA server.<br>Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches |
| Cisco IOS XE Fuji 16.8.1a | Change of Authorization | Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 33

# Configuring Identity Control Policies


- Information About Identity Control Policies, on page 637
- How to Configure Identity Control Policies, on page 640
- Configuration Examples for Cisco Identity-Based Control Policies, on page 653
- Feature History for Identity Control Policies, on page 657


# Information About Identity Control Policies

## Cisco Identity Based Networking Services Configuration

To convert all relevant authentication commands to their Class-Based Policy Language(CPL) control policy equivalents, use the **authentication convert-to new-style** command. This command permanently converts the legacy configuration on the switch to identity-based networking services.

**Note** This configuration is irreversible. It disables the conversion command – **authentication display [legacy | new-style]**.

Use the **authentication display config-mode** command in EXEC mode to display the current configuration mode; *legacy* if it is legacy mode and **new-style** if it is Identity-Based Networking Services configuration mode.

```
(Device)# authentication display config-mode
Current configuration mode is legacy

Device)# authentication display config-mode
Current configuration mode is new-style
```

## Concurrent Authentication Methods

Cisco IBNS allows the concurrent operation of IEEE 802.1x (dot1x), MAC authentication bypass (MAB), and web authentication methods, making it possible to invoke multiple authentication methods in parallel on a single subscriber session. This allows the client-supported method to complete at the earliest opportunity without the delays associated with serialization.

Typically, the access control method that is used to authorize a host is left up to the endpoint. For example, a printer without an 802.1x supplicant would be authorized through MAB only, an employee desktop through 802.1x only, and a guest through web authentication only. The default priority order is 802.1x, followed by MAB, then web authentication. When method priorities are the same, the first method that successfully authenticates the session prevails.

An example in which more than one method may succeed during the lifetime of a session is when MAB is used to provide interim access pending success of 802.1x. A host could be also be given interim access to a web server to allow credentials to be updated so that 802.1x can succeed after an authentication failure.

# Configuration Display Mode

Identity-Based Networking Services introduces new Cisco IOS commands that replace many of the previously supported authentication and policy commands. These commands are available only after enabling the Cisco common classification policy language (C3PL) display mode that supports Identity-Based Networking Services. Identity-Based Networking Services features such as concurrent authentication and web authentication with IPv6 are not supported in legacy mode.

The device defaults to the legacy configuration mode until you do one of the following:

- Enter the **authentication display new-style** command—This command switches to C3PL display mode, temporarily converting your legacy configuration to a Identity-Based Networking Services configuration so you can see how it looks before you make the conversion permanent. You can switch back to legacy mode by using the **authentication display legacy** command. See the Enabling the Display Mode for Cisco Identity Based Networking Services, on page 640 section.
- Enter a Identity-Based Networking Services configuration command—After you enter the first explicit Identity-Based Networking Services command, the configuration converts to C3PL display mode permanently and legacy commands are suppressed. The **authentication display** command is disabled and you can no longer revert to the legacy configuration mode.

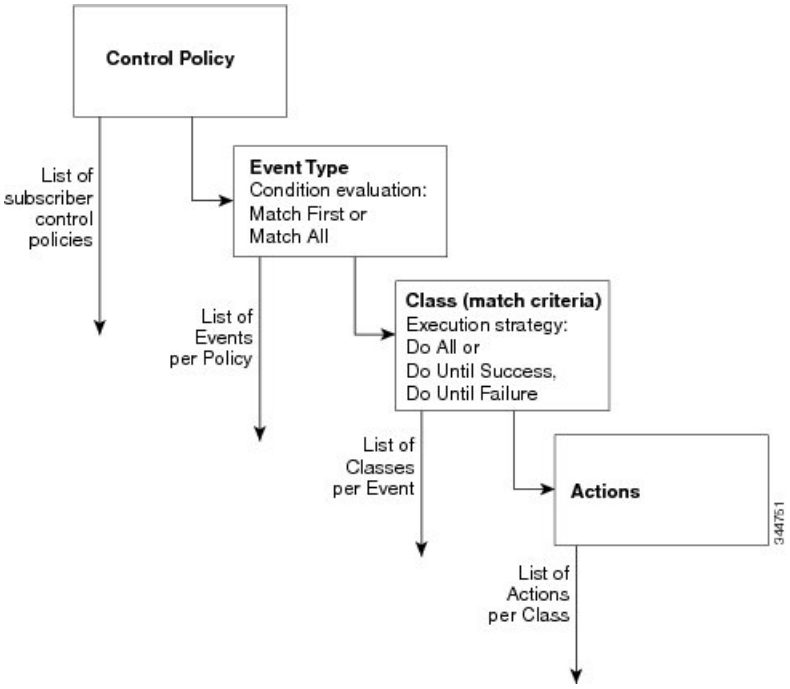
# Control Policies for Cisco Identity Based Networking Services

A control policy defines the handling of different subscriber life-cycle events. For various events, such as session start or session failure, you can specify actions in the control policy. These actions can be executed conditionally for different subscribers based on various match criteria. Control policies are activated on interfaces and typically control the authentication of subscriber identity and the activation of services on sessions. For example, you can configure a control policy to authenticate specific subscribers and then provide them with access to specific services.

A control policy consists of one or more control policy rules and a decision strategy that governs how the policy rules are evaluated. A control policy rule consists of a control class (a flexible condition clause), an event for which the condition is evaluated, and one or more actions. Actions are general system functions, such as authenticate or activate. You define the specific actions that an event will trigger and some events have default actions.

The figure below illustrates how each control policy contains a list of events that are considered applicable to the subscriber life cycle. Within each event type is a list of control classes with different match criteria for subscriber identity, and under each class is a list of actions to be executed.

*Figure 45: Control Policy Structure*




# Control Policy Configuration Overview

Control policies express system functionality in terms of an event, a condition, and an action. There are three steps in defining a control policy:

1. Create one or more control classes—A control class specifies the conditions that must be met for a control policy to be activated. A control class can contain multiple conditions, each of which will evaluate as either true or false. Match directives specify whether all, any, or none of the individual conditions must evaluate true for the class to evaluate true. Or, you can specify the default control class which does not contain any conditions and always evaluates true.
2. Create a control policy—A control policy contains one or more control policy rules. A control policy rule consists of a control class, an event that causes the class to be evaluated, and one or more actions. Actions are numbered and executed sequentially.
3. Apply the control policy—A control policy is activated by applying it to an interface.

# Parameter Maps for Cisco Identity Based Networking Services

A parameter map allows you to specify parameters that control the behavior of actions specified under a control policy. For Cisco IBNS, an authentication parameter map defines parameters used for the action specified with the **authenticate using webauth** command. You can configure the following types of parameter maps:

- Authentication bypass (This is also called nonresponsive host [NRH] authentication.)
- Consent

- Web authentication
- Web authentication with consent

Parameter maps are optional. If you do not configure a named parameter map, the software uses the default parameters that are specified in the global parameter map.

## Per User Inactivity Handling Across Methods

A common inactivity aging feature extends support for RADIUS attributes 28 (Idle-Timeout) and attribute 29 (Termination-Action) to web authenticated sessions, providing consistent inactivity handling across all authentication methods, including 802.1x, MAC authentication bypass (MAB), and web authentication. The AAA server sends these attributes as part of the user authorization. After a session has been idle for the amount of time specified in attribute 28, or has reached the timeout configured with attribute 29, the session is terminated.

You can also apply the inactivity timeout and absolute timeout to sessions through a locally defined service template. When enabling the inactivity timeout, you can also enable address resolution protocol (ARP) probes that are sent before the session is terminated.

# How to Configure Identity Control Policies

## Enabling the Display Mode for Cisco Identity Based Networking Services

Cisco IBNS features are configured in the Cisco common classification policy language (C3PL) display mode. The legacy authentication manager mode is enabled by default. You can use the following procedure to switch to C3PL display mode and temporarily convert any legacy configuration commands to their C3PL equivalents. This allows you to preview your legacy configuration as a Identity-Based Networking Services configuration before making the conversion permanent. After you enter an explicit Cisco IBNS command, the conversion becomes permanent and you can no longer revert to legacy mode.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **authentication display** {**legacy** \| **new-style**}<br>**Example:**<br>`Device# authentication display new-style` | Sets the display mode for authentication and policy configuration.<br>• The default display mode is legacy.<br>• You can use this command to switch between legacy and C3PL display mode until you execute the first explicit Identity-Based Networking Services command. After you enter the first explicit |

| | Command or Action | Purpose |
|---|---|---|
| | | Identity-Based Networking Services command, for example when configuring a control class or control policy, the system displays a prompt to confirm whether you want to continue because this command will be disabled and you cannot revert to legacy mode.<br><br>**Note**<br>If you save the configuration while the new-style mode is enabled, and then perform a reload, the display mode is permanently set to new-style. The **authentication display** command is disabled and you cannot revert to legacy mode.<br><br>For the stack devices and standalone devices to revert to legacy mode, save the new-style configuration in a flash, **write erase** the device and then perform a **reload**. |

# Configuring a Control Class

A control class defines the conditions under which the actions of a control policy are executed. You define whether all, any, or none of the conditions must evaluate true to execute the actions of the control policy. Control classes are evaluated based on the event specified in the control policy.

**Note** This procedure shows all of the match conditions that you can configure in a control class. You must specify at least one condition in a control class to make it valid. All other conditions, and their corresponding steps, are optional (steps 4 through 18 below).

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br><br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br><br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br><br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **class-map type control subscriber** {**match-all** \| **match-any** \| **match-none**} *control-class-name*<br>**Example:**<br>`Device(config)# class-map type control subscriber match-all DOT1X_NO_AGENT` | Creates a control class and enters control class-map filter mode.<br>• **match-all**—All of the conditions in the control class must evaluate true.<br>• **match-any**—At least one of the conditions in the control class must evaluate true.<br>• **match-none**—All of the conditions in the control class must evaluate false. |
| **Step 4** | {**match** \| **no-match**} **activated-service-template** *template-name*<br>**Example:**<br>`Device(config-filter-control-classmap)# match activated-service-template SVC_1` | (Optional) Creates a condition that evaluates true based on the service template activated on a session. |
| **Step 5** | {**match** \| **no-match**} **authorization-status** {**authorized** \| **unauthorized**}<br>**Example:**<br>`Device(config-filter-control-classmap)# match authorization-status authorized` | (Optional) Creates a condition that evaluates true based on a session's authorization status. |
| **Step 6** | {**match** \| **no-match**} **authorizing-method-priority** {**eq** \| **gt** \| **lt**} *priority-value*<br>**Example:**<br>`Device(config-filter-control-classmap)# match authorizing-method-priority eq 10` | (Optional) Creates a condition that evaluates true based on the priority of the authorization method.<br>• **eq**—Current priority is equal to *priority-value*.<br>• **gt**—Current priority is greater than *priority-value*.<br>• **lt**—Current priority is less than *priority-value*.<br>• *priority-value*—Priority value to match. Range: 1 to 254, where 1 is the highest priority and 254 is the lowest. |
| **Step 7** | {**match** \| **no-match**} **client-type** {**data** \| **switch** \| **video** \| **voice**}<br>**Example:**<br>`Device(config-filter-control-classmap)# match client-type data` | (Optional) Creates a condition that evaluates true based on an event's device type. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | {**match** \| **no-match**} **current-method-priority** {**eq** \| **gt** \| **lt**} *priority-value*<br>**Example:**<br>`Device(config-filter-control-classmap)# match current-method-priority eq 10` | (Optional) Creates a condition that evaluates true based on the priority of the current authentication method. |
| **Step 9** | {**match** \| **no-match**} **ip-address** *ip-address*<br>**Example:**<br>`Device(config-filter-control-classmap)# match ip-address 10.10.10.1` | (Optional) Creates a condition that evaluates true based on an event's source IPv4 address. |
| **Step 10** | {**match** \| **no-match**} **ipv6-address** *ipv6-address*<br>**Example:**<br>`Device(config-filter-control-classmap)# match ipv6-address FE80::1` | (Optional) Creates a condition that evaluates true based on an event's source IPv6 address. |
| **Step 11** | {**match** \| **no-match**} **mac-address** *mac-address*<br>**Example:**<br>`Device(config-filter-control-classmap)# match mac-address aabb.cc00.6500` | (Optional) Creates a condition that evaluates true based on an event's MAC address. |
| **Step 12** | {**match** \| **no-match**} **method** {**dot1x** \| **mab** \| **webauth**}<br>**Example:**<br>`Device(config-filter-control-classmap)# match method dot1x` | (Optional) Creates a condition that evaluates true based on an event's authentication method. |
| **Step 13** | {**match** \| **no-match**} **port-type** {**l2-port** \| **l3-port** \| **dot11-port**}<br>**Example:**<br>`Device(config-filter-control-classmap)# match port-type l2-port` | (Optional) Creates a condition that evaluates true based on an event's interface type. |
| **Step 14** | {**match** \| **no-match**} **result-type** [**method** {**dot1x** \| **mab** \| **webauth**}] *result-type*<br>**Example:**<br>`Device(config-filter-control-classmap)# match result-type agent-not-found` | (Optional) Creates a condition that evaluates true based on the specified authentication result.<br>• To display the available result types, use the question mark (**?**) online help function. |
| **Step 15** | {**match** \| **no-match**} **service-template** *template-name*<br>**Example:** | (Optional) Creates a condition that evaluates true based on an event's service template. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-filter-control-classmap)# match service-template svc_1` | |
| **Step 16** | {**match** \| **no-match**} **tag** *tag-name*<br>**Example:**<br>`Device(config-filter-control-classmap)# match tag tag_1` | (Optional) Creates a condition that evaluates true based on the tag associated with an event. |
| **Step 17** | {**match** \| **no-match**} **timer** *timer-name*<br>**Example:**<br>`Device(config-filter-control-classmap)# match timer restart` | (Optional) Creates a condition that evaluates true based on an event's timer. |
| **Step 18** | {**match** \| **no-match**} **username** *username*<br>**Example:**<br>`Device(config-filter-control-classmap)# match username josmiths` | (Optional) Creates a condition that evaluates true based on an event's username. |
| **Step 19** | **end**<br>**Example:**<br>`Device(config-filter-control-classmap)# end` | (Optional) Exits control class-map filter configuration mode and returns to privileged EXEC mode. |
| **Step 20** | **show class-map type control subscriber** {**all** \| **name** *control-class-name*}<br>**Example:**<br>`Device# show class-map type control subscriber all` | (Optional) Displays information about Identity-Based Networking Services control classes. |

### Example: Control Class

The following example shows a control class that is configured with two match conditions:

```
class-map type control subscriber match-all DOT1X_NO_AGENT
 match method dot1x
 match result-type agent-not-found
```

# Configuring a Control Policy

Control policies determine the actions that the system takes in response to specified events and conditions. The control policy contains one or more control policy rules that associate a control class with one or more actions. The actions that you can configure in a policy rule depend on the type of event that you specify.

**Note** This task includes all of the actions that you can configure in a control policy regardless of the event. All of these actions, and their corresponding steps, are optional (steps 6 through 21 below). To display the supported actions for a particular event, use the question mark (**?**) online help function.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **policy-map type control subscriber** *control-policy-name*<br>**Example:**<br>`Device(config)# policy-map type control subscriber POLICY_1` | Defines a control policy for subscriber sessions. |
| **Step 4** | **event** *event-name* [**match-all** \| **match-first**]<br>**Example:**<br>`Device(config-event-control-policymap)# event session-started` | Specifies the type of event that triggers actions in a control policy if conditions are met.<br>• **match-all** is the default behavior.<br>• To display the available event types, use the question mark (**?**) online help function. For a complete description of event types, see the **event** command. |
| **Step 5** | *priority-number* **class** {*control-class-name* \| **always**} [**do-all** \| **do-until-failure** \| **do-until-success**]<br>**Example:**<br>`Device(config-class-control-policymap)# 10 class always` | Associates a control class with one or more actions in a control policy.<br>• A named control class must first be configured before specifying it with the *control-class-name* argument.<br>• **do-until-failure** is the default behavior. |
| **Step 6** | *action-number* **activate** {**policy type control subscriber** *control-policy-name* [**child** [**no-propagation** \| **concurrent**] \| **service-template** *template-name* [**aaa-list** *list-name*] [**precedence** *number*] [**replace-all**]}<br>**Example:**<br>`Device(config-action-control-policymap)# 10 activate service-template FALLBACK` | (Optional) Activates a control policy or service template on a subscriber session. |
| **Step 7** | *action-number* **authenticate using** {**dot1x** \| **mab** \| **webauth**} [**aaa** {**authc-list** *authc-list-name* \| **authz-list** *authz-list-name*]} [**merge**] [**parameter-map** *map-name*] | (Optional) Initiates the authentication of a subscriber session using the specified method. |

| | Command or Action | Purpose |
|---|---|---|
| | [**priority** *priority-number*] [**replace** \| **replace-all**] [**retries** *number* {**retry-time** *seconds*}]<br>**Example:**<br>`Device(config-action-control-policymap)# 10 authenticate using dot1x priority 10` | |
| **Step 8** | *action-number* **authentication-restart** *seconds*<br>**Example:**<br>`Device(config-action-control-policymap)# 20 authentication-restart 60` | (Optional) Sets a timer to restart the authentication process after an authentication or authorization failure. |
| **Step 9** | *action-number* **authorize**<br>**Example:**<br>`Device(config-action-control-policymap)# 10 authorize` | (Optional) Initiates the authorization of a subscriber session. |
| **Step 10** | *action-number* **clear-authenticated-data-hosts-on-port**<br>**Example:**<br>`Device(config-action-control-policymap)# 20 clear-authenticated-data-hosts-on-port` | (Optional) Clears authenticated data hosts on a port after an authentication failure. |
| **Step 11** | *action-number* **clear-session**<br>**Example:**<br>`Device(config-action-control-policymap)# 30 clear-session` | (Optional) Clears an active subscriber session. |
| **Step 12** | *action-number* **deactivate** {**policy type control subscriber** *control-policy-name* \| **service-template** *template-name*}<br>**Example:**<br>`Device(config-action-control-policymap)# 20 deactivate service-template interface_template` | (Optional) Deactivates a control policy or service template on a subscriber session. |
| **Step 13** | *action-number* **err-disable**<br>**Example:**<br>`Device(config-action-control-policymap)# 10 err-disable` | (Optional)Temporarily disables a port after a session violation event. |
| **Step 14** | *action-number* **pause reauthentication**<br>**Example:**<br>`Device(config-action-control-policymap)# 20 pause reauthentication` | (Optional) Pauses reauthentication after an authentication failure. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 15** | *action-number* **protect**<br>**Example:**<br>`Device(config-action-control-policymap)# 10 protect` | (Optional) Silently drops violating packets after a session violation event. |
| **Step 16** | *action-number* **replace**<br>**Example:**<br>`Device(config-action-control-policymap)# 10 replace` | (Optional) Clears the existing session and creates a new session after a violation event. |
| **Step 17** | *action-number* **restrict**<br>**Example:**<br>`Device(config-action-control-policymap)# 10 restrict` | (Optional) Drops violating packets and generates a syslog entry after a session violation event. |
| **Step 18** | *action-number* **resume reauthentication**<br>**Example:**<br>`Device(config-action-control-policymap)# 20 resume reauthentication` | (Optional) Resumes the reauthentication process after an authentication failure. |
| **Step 19** | *action-number* **set-timer** *timer-name seconds*<br>**Example:**<br>`Device(config-action-control-policymap)# 20 set-timer RESTART 60` | (Optional) Starts a named policy timer. |
| **Step 20** | *action-number* **terminate** {**dot1x** \| **mab** \| **webauth**}<br>**Example:**<br>`Device(config-action-control-policymap)# 20 terminate webauth` | (Optional) Terminates an authentication method on a subscriber session. |
| **Step 21** | *action-number* **unauthorize**<br>**Example:**<br>`Device(config-action-control-policymap)# 20 unauthorize` | (Optional) Removes all authorization data from a subscriber session. |
| **Step 22** | **end**<br>**Example:**<br>`Device(config-action-control-policymap)# end` | (Optional) Exits control policy-map action configuration mode and returns to privileged EXEC mode. |
| **Step 23** | **show policy-map type control subscriber** {**all** \| **name** *control-policy-name*}<br>**Example:**<br>`Device# show policy-map type control subscriber POLICY_1` | (Optional) Displays information about identity control policies. |

### Example: Control Policy

The following example shows a simple control policy with the minimum configuration necessary for initiating authentication:

```
policy-map type control subscriber POLICY_1
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using dot1x
```

For detailed examples of control policies for concurrent and sequential authentication, see the Configuration Examples for Cisco Identity-Based Control Policies, on page 653 section.

# Applying a Control Policy to an Interface

Control policies typically control the authentication of subscriber identity and the activation of services on sessions. Perform this task to apply a control policy to an interface.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface GigabitEthernet 1/0/1` | Specifies an interface and enters interface configuration mode. |
| **Step 4** | **service-policy type control subscriber** *control-policy-name*<br>**Example:**<br>`Device(config-if)# service-policy type control subscriber POLICY_1` | Applies a previously configured control policy.<br>• To display a list of all configured control policies, use the question mark (**?**) online help function. |
| **Step 5** | **subscriber aging** {**inactivity-timer** *seconds* [**probe**] \| **probe**}<br>**Example:**<br>`Device(config-if)# subscriber aging inactivity-timer 60 probe` | Enables an inactivity timer for subscriber sessions.<br>Starting with Cisco IOS XE Everest 16.5.1a if you configure this command, you must also configure the device-tracking binding reachable-lifetime command in global configuration mode, for probes to work as |

| | Command or Action | Purpose |
|---|---|---|
| | | expected. Configure a reachable lifetime with the same value as the inactivity timer probe. This way, when the reachable lifetime expires, the state of the entry changes based on the reachability of the host. For more information, see the device-tracking binding command in the command reference of the corresponding release. |

#### Example: Applying a Control Policy to an Interface

```
interface GigabitEthernet 1/0/2
 subscriber aging inactivity-timer 60 probe
 device-tracking binding reachable-lifetime 60
 service-policy type control subscriber POLICY_1
```

## Configuring Authentication Features on Ports

Perform this task to control access to a port, including the port authorization state, host access mode, preauthentication access, and the authentication direction.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/2` | Enters interface configuration mode for the selected interface. |
| **Step 4** | **access-session port-control** {**auto** \| **force-authorized** \| **force-unauthorized**}<br>**Example:**<br>`Device(config-if)# access-session port-control auto` | Sets the authorization state of a port.<br>• The default value is **force-authorized**. |
| **Step 5** | **access-session host-mode** {**multi-auth** \| **multi-domain** \| **multi-host** \| **single-host**} | Allows hosts to gain access to a controlled port. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device(config-if)# access-session host-mode single-host` | • To use this command, you must first enable the **access-session port-control auto** command.<br>• The default value is **multi-auth**. |
| **Step 6** | **access-session closed**<br>**Example:**<br>`Device(config-if)# access-session closed` | Prevents preauthentication access on this port.<br>• The port is set to open access by default. |
| **Step 7** | **access-session control-direction** {**both** \| **in**}<br>**Example:**<br>`Device(config-if)# access-session control-direction in` | Sets the direction of authentication control on a port.<br>• The default value is **both**. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 9** | **show access-session interface** *interface-type interface-number* [**details**]<br>**Example:**<br>`Device# show access-session interface gigabitethernet 1/0/2 details` | Displays information about subscriber sessions that match the specified client interface. |

#### Example: Port Authentication

```
interface GigabitEthernet 1/0/2
 access-session host-mode single-host
 access-session closed
 access-session port-control auto
 access-session control-direction in
```

## Configuring a Parameter Map for Web-Based Authentication

A parameter map allows you to modify parameters that control the behavior of actions configured under a control policy. A parameter map for web-based authentication sets parameters that can be applied to subscriber sessions during authentication. If you do not create a parameter map, the policy uses default parameters.

Perform the following steps to define either a global or named parameter map for web-based authentication.

**Note** The configuration commands available in the global parameter map differ from the commands available in a named parameter map.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **parameter-map type webauth** {*parameter-map-name* \| **global**}<br>**Example:**<br>`Device(config)# parameter-map type webauth MAP_2` | Creates a parameter map and enters parameter-map webauth configuration mode.<br>• The specific configuration commands supported for a global parameter map defined with the **global** keyword differ from the commands supported for a named parameter map defined with the *parameter-map-name* argument. |
| **Step 4** | **banner** {**file** *location*:*filename* \| **text** *banner-text*}<br>**Example:**<br>`Device(config-params-parameter-map)# banner file flash:webauth_banner.html` | (Optional) Displays a banner on the web-authentication login web page. |
| **Step 5** | **consent**<br>**Example:**<br>`Device(config-params-parameter-map)# type consent` | (Optional) Defines the methods supported by a web-based authentication parameter map.<br>• This command is supported in named parameter maps only. |
| **Step 6** | **consent email**<br>**Example:**<br>`Device(config-params-parameter-map)# consent email` | (Optional) Requests a user's e-mail address on the web-authentication login web page.<br>• This command is supported in named parameter maps only. |
| **Step 7** | **custom-page** {**failure** \| **login** [**expired**] \| **success**} **device** *location*:*filename*<br>**Example:**<br>`Device(config-params-parameter-map)# custom-page login device flash:webauth_login.html`<br>`Device(config-params-parameter-map)# custom-page login expired device flash:webauth_expire.html`<br>`Device(config-params-parameter-map)# custom-page success device` | (Optional) Displays custom authentication proxy web pages during web-based authentication.<br>• You must configure all four custom HTML files. If fewer than four files are configured, the internal default HTML pages will be used. |

| | Command or Action | Purpose |
|---|---|---|
| | `flash:webauth_success.html`<br>`Device(config-params-parameter-map)# custom-page failure device flash:webauth_fail.html` | |
| **Step 8** | **max-http-conns** *number*<br>**Example:**<br>`Device(config-params-parameter-map)# max-http-conns 5` | (Optional) Limits the number of HTTP connections for each web authentication client. |
| **Step 9** | **redirect** {{**for-login** \| **on-failure** \| **on-success**} *url* \| **portal** {**ipv4** *ipv4-address* \| **ipv6** *ipv6-address*}}<br>**Example:**<br>`Device(config-params-parameter-map)# redirect portal ipv6 FE80::1`<br>`Device(config-params-parameter-map)# redirect on-failure http://10.10.3.34/~sample/failure.html` | (Optional) Redirects users to a particular URL during web-based authentication. |
| **Step 10** | **timeout init-state sec** *seconds*<br>**Example:**<br>`Device(config-params-parameter-map)# timeout init-state sec 60` | (Optional) Sets the Init state timeout for web-based authentication sessions.<br>• The range of seconds is (60-3932100). |
| **Step 11** | **type** {**authbypass** \| **consent** \| **webauth** \| **webconsent**}<br>**Example:**<br>`Device(config-params-parameter-map)# type consent` | (Optional) Defines the methods supported by a web-based authentication parameter map.<br>• This command is supported in named parameter maps only. |
| **Step 12** | **virtual-ip** {**ipv4** *ipv4-address* \| **ipv6** *ipv6-address*}<br>**Example:**<br>`Device(config-params-parameter-map)# virtual-ip ipv6 FE80::1` | (Optional) Specifies a virtual IP address for web-based authentication clients.<br>• This command is supported in the global parameter map only. |
| **Step 13** | **watch-list** {**add-item** {**ipv4** *ipv4-address* \| **ipv6** *ipv6-address*} \| **dynamic-expiry-timeout** *minutes* \| **enabled**}<br>**Example:**<br>`Device(config-params-parameter-map)# watch-list enabled`<br>`Device(config-params-parameter-map)# watch-list dynamic-expiry-timeout 20`<br>`Device(config-params-parameter-map)# watch-list add-item ipv6 FE80::1` | (Optional) Enables a watch list of web-based authentication clients.<br>• This command is supported in the global parameter map only. |
| **Step 14** | **end**<br>**Example:** | (Optional) Exits parameter-map configuration mode and returns to privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-params-parameter-map)# end` | |
| **Step 15** | **show ip admission status** [**banners** \| **custom-pages** \| **parameter-map** [*parameter-map*]]<br>**Example:**<br>`Device# show ip admission status custom-pages` | (Optional) Displays information about configured banners and custom pages. |

#### Example: Parameter Map for Web-Based Authentication

```
parameter-map type webauth PMAP_2
 type webconsent
 timeout init-state sec 60
 max-http-conns 5
 type consent
 consent email
 custom-page login device flash:webauth_login.html
 custom-page success device flash:webauth_success.html
 custom-page failure device flash:webauth_fail.html
 custom-page login expired device flash:webauth_expire.html
```

#### What to do next

Apply the parameter map to sessions by specifying it in the **authenticate using** command when configuring a Control Policy. See the Configuring a Control Policy, on page 644 section.

# Configuration Examples for Cisco Identity-Based Control Policies

## Example: Configuring Control Policy for Concurrent Authentication Methods

The following example shows a control policy that is configured to allow concurrent authentication. All three methods (dot1x, MAB, and web authentication) are run simultaneously when a session is started. The dot1x method is set to the highest priority and web authentication has the lowest priority, which means that if multiple methods succeed, the highest priority method is honored.

If authentication fails, the session manager checks whether all methods have failed, and if so, it sets the restart timer to 60 seconds, after which it attempts to start all three methods again. On authentication success, the session manager terminates any lower priority methods; for dot1x, this is MAB and webauth; for MAB it is webauth. Lastly, if session manager detects a dot1x client (agent-found) it triggers only dot1x to run.

The class map named ALL-FAILED checks that all three methods have run to completion (result type is none until then) and that none of them was successful. In other words, all three methods have completed and failed.

**Note** When configuring a control policy for concurrent authentication, you must include a policy rule that explicitly terminates one method after another method of a higher priority succeeds.

```
class-map type control subscriber match-all ALL_FAILED
 no-match result-type method dot1x none
 no-match result-type method dot1x success
 no-match result-type method mab none
 no-match result-type method mab success
 no-match result-type method webauth none
 no-match result-type method webauth success
!
class-map type control subscriber match-all DOT1X
 match method dot1x
!
class-map type control subscriber match-all MAB
 match method mab
!
policy-map type control subscriber CONCURRENT_DOT1X_MAB_WEBAUTH
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using mab priority 20
   20 authenticate using dot1x priority 10
   30 authenticate using webauth parameter-map WEBAUTH_DEFAULT priority 30
 event authentication-failure match-first
  10 class ALL_FAILED
   10 authentication-restart 60
 event authentication-success match-all
  10 class DOT1X
   10 terminate MAB
   20 terminate webauth
  20 class MAB
   10 terminate webauth
 event agent-found match-all
  10 class always do-until-failure
   10 authenticate using dot1x priority 10
```

# Example: Configuring Control Policy for Sequential Authentication Methods

The following example shows a control policy that is configured to allow sequential authentication methods using 802.1X (dot1x), MAB, and web authentication.

```
parameter-map type webauth WEBAUTH_FALLBACK
 type webauth
!
class-map type control subscriber match-all DOT1X_NO_RESP
 match method dot1x
 match result-type method dot1x agent-not-found
!
class-map type control subscriber match-all MAB_FAILED
 match method mab
 match result-type method mab authoritative
!
policy-map type control subscriber POLICY_Gi3/0/10
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using dot1x priority 10
```

```
 event authentication-failure match-first
  10 class DOT1X_NO_RESP do-until-failure
   10 terminate dot1x
   20 authenticate using mab priority 20
  20 class MAB_FAILED do-until-failure
   10 terminate mab
   20 authenticate using webauth parameter-map WEBAUTH_FALLBACK priority 30
  30 class always do-until-failure
   10 terminate dot1x
   20 terminate mab
   30 terminate webauth
   40 authentication-restart 60
 event agent-found match-all
  10 class always do-until-failure
   10 terminate mab
   20 terminate webauth
   30 authenticate using dot1x priority 10
```

The following example shows a control policy that is configured to allow sequential authentication methods using 802.1X and MAB. If authentication fails, a service template for VLAN is activated.

```
service-template VLAN210
 vlan 210
!
class-map type control subscriber match-all DOT1X_FAILED
 match method dot1x
 match result-type method dot1x authoritative
!
class-map type control subscriber match-all DOT1X_NO_RESP
 match method dot1x
 match result-type method dot1x agent-not-found
!
class-map type control subscriber match-all MAB_FAILED
 match method mab
 match result-type method mab authoritative
!
policy-map type control subscriber POLICY_Gi3/0/14
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using dot1x retries 2 retry-time 0 priority 10
 event authentication-failure match-first
  10 class DOT1X_NO_RESP do-until-failure
   10 terminate dot1x
   20 authenticate using mab priority 20
  20 class MAB_FAILED do-until-failure
   10 terminate mab
   20 activate service-template VLAN210
   30 authorize
  30 class DOT1X_FAILED do-until-failure
   10 terminate dot1x
   20 authenticate using mab priority 20
  40 class always do-until-failure
   10 terminate dot1x
   20 terminate mab
   30 authentication-restart 60
 event agent-found match-all
  10 class always do-until-failure
   10 terminate mab
   20 authenticate using dot1x retries 2 retry-time 0 priority 10
```

# Example: Configuring Parameter Maps

### Global Parameter Map

The following example shows the configuration of a global parameter map:

```
parameter-map type webauth global
 timeout init-state sec 15
 watch-list enabled
 virtual-ip ipv6 FE80::1
 redirect on-failure http://10.10.3.34/~sample/failure.html
 max-http-conns 100
 watch-list dynamic-expiry-timeout 5000
 banner file flash:webauth_banner.html
```

### Named Parameter Maps for Web Authentication and Authentication Bypass (nonresponsive host [NRH])

The following example shows the configuration of two named parameter maps; one for web authentication and one for authentication bypass. This example also shows the corresponding control policy configuration.

```
parameter-map type webauth WEBAUTH_BANNER
 type webauth
 banner
!
parameter-map type webauth WEBAUTH_NRH
 type authbypass
!
class-map type control subscriber match-all NRH_FAIL
 match method webauth
 match current-method-priority eq 254
!
policy-map type control subscriber WEBAUTH_NRH
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using webauth parameter-map WEBAUTH_NRH priority 254
 event authentication-failure match-all
  10 class NRH_FAIL do-until-failure
   10 terminate webauth
   20 authenticate using webauth parameter-map WEBAUTH_BANNER priority 30
```

### Named Parameter Map for Web Authentication Using Custom Pages

The following example shows the configuration of a named parameter map for web authentication that defines custom pages for the login process, along with a control policy that uses the parameter map.

```
parameter-map type webauth CUSTOM_WEBAUTH
 type webauth
 custom-page login device flash:login_page.htm
 custom-page success device flash:success_page.htm
 custom-page failure device flash:fail_page.htm
 custom-page login expired device flash:expire_page.htm
!
policy-map type control subscriber CUSTOM_WEBAUTH
 event session-started match-all
```

```
  10 class always do-until-failure
   10 authenticate using webauth parameter-map CUSTOM_WEB retries 2 retry-time 0
```

### Named Parameter Map for Consent

The following example shows the configuration of a named parameter map for consent, along with the corresponding control policy that uses the parameter map:

```
parameter-map type webauth CONSENT
 type consent
!
ip access-list extended GUEST_ACL
 permit ip any 172.30.30.0 0.0.0.255
 permit ip any host 172.20.249.252
!
service-template GUEST_POLICY
 access-group GUEST_ACL
!
policy-map type control subscriber CONSENT
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using webauth parameter-map CONSENT
 event authentication-success match-all
  10 class always do-until-failure
   10 activate service-template GUEST_POLICY
```

### Named Parameter Map for Web Authentication with Consent

The following example shows the configuration of a named parameter map for web authentication with consent, along with the corresponding control policy that uses the parameter map:

```
parameter-map type webauth WEBAUTH_CONSENT
 type webconsent
!
ip access-list extended GUEST_ACL
 permit ip any 172.30.30.0 0.0.0.255
 permit ip any host 172.20.249.252
!
service-template GUEST_POLICY
 access-group GUEST_ACL
!
policy-map type control subscriber WEBAUTH_CONSENT
 event session-started match-all
  10 class always do-until-failure
   10 authenticate using webauth parameter-map CONSENT
 event authentication-success match-all
  10 class always do-until-failure
   10 activate service-template GUEST_POLICY
```

# Feature History for Identity Control Policies

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

*Table 36: Feature History for Identity Control Policies*

| Release | Feature Name | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Cisco Common Classification Policy Language-Based Identity Configuration | Identity control policies define the actions taken in response to specified events and conditions.<br>Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches |
| Cisco IOS XE Fuji 16.8.1a | Cisco Common Classification Policy Language-Based Identity Configuration | Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 34

# Policy Classification Engine

The Policy Classification Engine feature helps configure device-based policies and client (network endpoint) profiling and enforces a per user or per device policy on a network. The policy classification engine enables bring-your-own-device (BYOD) deployments integrate user or wireless device policies into the wireless controller. This module explains how to configure policies and apply them to a wireless LAN (WLAN).


- Restrictions for Policy Classification Engine, on page 659
- Information About Policy Classification Engine, on page 659
- How to Configure Policy Classification Engine, on page 660
- Configuration Examples for Policy Classification Engine, on page 663
- Feature History for Policy Classification Engine, on page 663


# Restrictions for Policy Classification Engine

Interface templates are not valid on wireless sessions.

# Information About Policy Classification Engine

## Policy Classification Engine Overview

The Policy Classification Engine feature helps configure device-based policies and client (network endpoint) profiling and enforces a per user or per device policy on a network.

You can configure sets of different policies that can be used for lookup and sequential matching. A policy is matched based on the configured policy statement. Use policies to profile devices based on the Dynamic Host Control Protocol (DHCP) or HTTP to identify end devices in a network. You can enforce specific policies at network endpoints.

The device (switch; for example, Cisco Catalyst 3850 Wireless LAN Controller) uses these attributes and predefined classification profiles to identify devices.

Policies are configured based on the following parameters:

- Device—Types of end devices. Examples are Windows machines, smart phones, Apple device like iPads, iPhones, and so on.
- Regular expressions

- User role—The user type or user group to which an user belongs. Examples are students, employees, and so on.
- Username—Login credentials entered by users.
- Time-of-day—The time-of-day when endpoints are allowed into a network.
- OUI—The MAC address that identifies the Organizational Unique Identifier (OUI).
- MAC address—The MAC address of the endpoint.

Once the device (switch) has a match corresponding to the policy parameters per end point, a policy is added. Policy enforcement is based on the following session attributes:

- VLAN—User-defined VLAN
- Access control list (ACL)
- Session timeout value—User-defined timeout for client sessions
- Quality of service (QoS)

You can configure policies and based on the session attributes, enforce these policies on end points.

# How to Configure Policy Classification Engine

## Configuring Policies in Cisco Identity Based Networking Services

To configure policies, perform the following tasks:

1. Configure a service template.
2. Configure an interface template.
3. Create a parameter map.
4. Create a policy map.
5. Apply the policy on a wireless LAN (WLAN).

## Configuring a Subscriber Parameter Map

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **parameter-map type subscriber attribute-to-service** *parameter-map-name*<br>**Example:**<br>`Device(config)# parameter-map type subscriber attribute-to-service param-map` | Configures a subscriber parameter map and enters parameter-map filter configuration mode. |
| **Step 4** | *priority-number* **map device-type eq** *device-type* **oui eq** *MAC-address*<br>**Example:**<br>`Device(config-parameter-map-filter)# 1 map device-type eq "Cisco-IP-Phone-9971" oui eq "08.cc.68"` | Maps the priority and the Organizationally Unique Identifier (OUI) of the configured device, and enters parameter-map filter submode configuration mode. |
| **Step 5** | *action-number* **interface-template** *interface-template-name*<br>**Example:**<br>`Device(config-parameter-map-filter-submode)# 2 interface-template IP-PHONE-INTERFACE-TEMPLATE` | Maps the action number to an interface template. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-parameter-map-filter-submode)# end` | Exits parameter-map filter submode configuration mode and returns to privileged EXEC mode. |
| **Step 7** | **show parameter-map type subscriber attribute-to-service** *parameter-map-name*<br>**Example:**<br>`Device# show parameter-map type subscriber attribute-to-service parameter-map-name` | Displays information about the specified parameter map. |

### Example

The following is sample output from the **show parameter-map type subscriber attribute-to-service** command:

```
Device# show parameter-map type subscriber attribute-to-service param-map

Parameter-map name: param-map
 Map: 1 map device-type eq "Cisco-IP-Phone-9971" oui eq "08.cc.68"
  Action(s):
   2 interface-template IP-PHONE-INTERFACE-TEMPLATE
```

# Configuring a Subscriber Policy Map

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **policy-map type control subscriber** *policy-map-name*<br>**Example:**<br>`Device(config)# policy-map type control subscriber pmap` | Defines a control policy for subscriber sessions and enters control policy-map event configuration mode. |
| **Step 4** | **event identity-update** {**match-all** \| **match-first**}<br>**Example:**<br>`Device(config-event-control-policymap)# event identity-update match-all` | Specifies the event type that triggers actions in a control policy if conditions are met, and enters control policy-map class configuration mode. |
| **Step 5** | *priority-number* **class always** {**do-all** \| **do-until-failure** \| **do-until-success**}<br>**Example:**<br>`Device(config-class-control-policymap)# 1 class always do-until-failure` | Associates a control class with one or more actions in a control policy and enters control policy-map action configuration mode. |
| **Step 6** | *action-number* **map attribute-to-service table** *parameter-map-name*<br>**Example:**<br>`Device(config-action-control-policymap)# 2 map attribute-to-service table param-map` | Maps identity-update attribute to an autoconf template. |
| **Step 7** | **end**<br>**Example:**<br>`Device(config-action-control-policymap)# end` | Exits control policy-map action configuration mode and returns to privileged EXEC mode. |
| **Step 8** | **show policy-map type control subscriber** *policy-map-name*<br>**Example:** | Displays information and statistics about the control policies. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# show policy-map type control subscriber pmap` | |

### Example

The following is sample output from the **show policy-map type control subscriber** command:

```
Device# show policy-map type control subscriber pmap

show policy-map type control subscriber pmap
policy-map
  event identity-update match-all
    1 class always do-until-failure
      1 map attribute-to-service table param-map
```

# Configuration Examples for Policy Classification Engine

## Example: Configuring a Subscriber Parameter Map

```
Device# configure terminal
Device(config)# parameter-map type subscriber attribute-to-service param-map
Device(config-parameter-map-filter)# 1 map device-type eq "Cisco-IP-Phone-9971" oui "eq
08.cc.68"
Device(config-parameter-map-filter-submode)# 2 interface-template IP-PHONE-INTERFACE-TEMPLATE
Device(config-parameter-map-filter-submode)# end
```

## Example: Configuring a Subscriber Policy Map

```
Device# configure terminal
Device(config)# policy-map type control subscriber pmap
Device(config-event-control-policymap)# event identity-update match-all
Device(config-class-control-policymap)# 1 class always do-until-failure
Device(config-action-control-policymap)# 2 map attribute-to-service table param-map
Device(config-action-control-policymap)# end
```

# Feature History for Policy Classification Engine

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

*Table 37: Feature History for Policy Classification Engine*

| Release | Feature Name | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Policy Classification Engine | The Policy Classification Engine feature helps configure device-based policies and client (network endpoint) profiling and enforces a per user or per device policy on a network. The policy classification engine enables bring-your-own-device (BYOD) deployments integrate user or wireless device policies into the wireless controller.<br><br>Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches |
| Cisco IOS XE Fuji 16.8.1a | Policy Classification Engine | Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 35

# Configuring Identity Service Templates

Identity service templates contain a set of policy attributes or features that can be applied to one or more subscriber sessions through a control policy, a RADIUS Change of Authorization (CoA) request, or a user profile or service profile. This module provides information about how to configure local service templates for Identity-Based Networking Services.


- Prerequisites for Identity Service Templates, on page 665
- Information About Identity Service Templates, on page 665
- How to Configure Identity Service Templates, on page 666
- Configuration Examples for Identity Service Templates, on page 668
- Feature History for Identity Service Templates, on page 670


# Prerequisites for Identity Service Templates

For downloadable service templates, the switch uses the default password “cisco123” when downloading the service templates from the authentication, authorization, and accounting (AAA) server, Cisco Secure Access Control Server (ACS), or Cisco Identity Services Engine (ISE). The AAA, ACS, and ISE server must include the password “cisco123” in the service template configuration.

# Information About Identity Service Templates

## Service Templates for Cisco Identity-Based Networking Services

A service template contains a set of service-related attributes or features, such as access control lists (ACLs) and VLAN assignments, that can be activated on one or more subscriber sessions in response to session life-cycle events. Templates simplify the provisioning and maintenance of network session policies where policies fall into distinct groups or are role-based.

A service template is applied to sessions through its reference in a control policy, through RADIUS Change of Authorization (CoA) requests, or through a user profile or service profile. User profiles are defined per subscriber; service profiles can apply to multiple subscribers.

Identity-Based Networking Services supports two types of service templates:

- Downloadable Service Templates—The service template is configured centrally on an external ACS or AAA server and downloaded on demand.

- Locally Configured Service Templates—The service template is configured locally on the device through the Cisco IOS command-line interface (CLI).

## Downloadable Service Templates

Cisco Identity Based Networking Services (IBNS) can download a service template defined on an external AAA server. The template defines a collection of AAA attributes. These templates are applied to sessions through the use of vendor-specific attributes (VSAs) included in RADIUS CoA messages received from the external AAA server or ACS. The name of the template is referenced in a user profile or a control policy, which triggers a download of the service template during processing.

The downloadable template is cached on the device and subsequent requests for a download will refer to the available cached template. The template however is cached only for the duration of its active usage. The downloaded template cached on the device is protected and cannot be deleted through the command line interface or through other applications. This ensures that the template is deleted only when there are no active references to it.

## Locally Configured Service Templates

Service templates can be configured locally through the CLI. These service templates can be applied to subscriber sessions by a reference in a control policy.

When an active local template is updated, changes to that local template will be reflected across all sessions for which the template is active. If a template is deleted, all content from that template that is applied against sessions is removed.

# How to Configure Identity Service Templates

## Configuring a Local Service Template

A service template defines the local policies that can be applied to a subscriber session. Activate this service template on sessions on which the local policies must be applied.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **service-template** *template-name*<br>**Example:**<br>`Device(config)# service-template SVC_2` | Creates a service template and enters service template configuration mode. |
| **Step 4** | **absolute-timer** *minutes*<br>**Example:**<br>`Device(config-service-template)# absolute-timer 15` | (Optional) Enables an absolute timeout for subscriber sessions. |
| **Step 5** | **access-group** *access-list-name*<br>**Example:**<br>`Device(config-service-template)# access-group ACL_2` | (Optional) Applies an access list to sessions using a service template. |
| **Step 6** | **description** *description*<br>**Example:**<br>`Device(config-service-template)# description label for SVC_2` | (Optional) Adds a description for a service template. |
| **Step 7** | **inactivity-timer** *minutes* [**probe**]<br>**Example:**<br>`Device(config-service-template)# inactivity-timer 15` | (Optional) Enables an inactivity timeout for subscriber sessions. |
| **Step 8** | **redirect url** *url*<br>**Example:**<br>`Device(config-service-template)# redirect url www.cisco.com` | (Optional) Redirects clients to a particular URL. |
| **Step 9** | **sgt** *range*<br>**Example:**<br>`Device(config-service-template)# sgt 100` | (Optional) Associates a Security Group Tag (SGT) with a service template. |
| **Step 10** | **tag** *tag-name*<br>**Example:**<br>`Device(config-service-template)# tag TAG_2` | (Optional) Associates a user-defined tag with a service template. |
| **Step 11** | **vlan** *vlan-id*<br>**Example:**<br>`Device(config-service-template)# vlan 215` | (Optional) Applies a VLAN to sessions using a service template. |
| **Step 12** | **end**<br>**Example:**<br>`Device(config-service-template)# end` | Exits service template configuration mode and returns to privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 13** | **show service-template** [*template-name*]<br>**Example:**<br>`Device# show service-template SVC_2` | Displays information about configured service templates. |

### Example: Service Template

```
service-template SVC_2
 description label for SVC_2
 access-group ACL_2
 redirect url www.cisco.com
 vlan 215
 inactivity-timer 15
 absolute-timer 15
 tag TAG_2
```

### What to do next

To activate a service template on a subscriber session, specify the service template in a control policy. See Configuring a Control Policy, on page 644.

# Configuration Examples for Identity Service Templates

## Example: Activating a Service Template and Replace All

### Local Service Template Configuration

The following example shows the configuration of a service template defined locally on the device. This template contains attributes that are applied to sessions that use the control policy named POSTURE_VALIDATION, shown below:

```
service-template DOT1X
 access-group SVC1_ACL
 redirect url www.cisco.com match URL_REDIRECT_ACL
 inactivity-timer 60
 absolute-timer 300
!
ip access-list extended URL_REDIRECT_ACL
 permit tcp any host 5.5.5.5 eq www
```

### Control Policy Configuration

The following example shows a control policy that activates the service template named DOT1X with replace-all enabled. The successfully activated template will replace the existing authorization data and any service template previously applied to the session.

```
policy-map type control subscriber POSTURE_VALIDATION
 event session-started match-all
  10 class always do-until-failure
```

```
    10 authenticate using dot1x priority 10
    20 authenticate using webauth priority 20
  event authentication-success match-all
   10 class DOT1X do-all
    10 terminate webauth
    20 activate service-template DOT1X replace-all
```

# Example: Activating a Service Template for Fallback Service

### Local Service Template Configuration

The following example shows the configuration of a service template defined locally on the device. This template contains attributes that are applied to sessions that use the control policy named POSTURE_VALIDATION, shown below:

```
service-template FALLBACK
 description fallback service
 access-group ACL_2
 redirect url www.cisco.com
 inactivity-timer 15
 absolute-timer 15
 tag TAG_2
```

### Control Policy Configuration

The following example shows a control policy that runs authentication methods dot1x and MAB. If dot1x authentication fails, MAB authentication is attempted. If MAB fails, the system provides a default authorization profile using the FALLBACK template.

```
policy-map type control subscriber POSTURE_VALIDATION
 event session-started match-all
  10 class always do-all
   10 authenticate using dot1x
 event authentication-failure match-all
  10 class DOT1X do-all
   10 authenticate using mab
  20 class MAB do-all
   10 activate service-template FALLBACK
```

# Example: Deactivating a Service Template

### Access Control List Configuration

The following example shows the configuration of an access control list (ACL) that is used by the local service template named LOW_IMPACT_TEMPLATE, shown below.

```
ip access-list extended LOW_IMPACT_ACL
 permit udp any any eq bootps
 permit tcp any any eq www
 permit tcp any any eq 443
 permit ip any 172.30.0.0 0.0.255.255
```

### Local Service Template Configuration

The following example shows the configuration of the local service template that provides limited access to all hosts even when authentication fails.

```
service-template LOW_IMPACT_TEMPLATE
 description Service template for Low impact mode
 access-group LOW_IMPACT_ACL
 inactivity-timer 60
 tag LOW_IMPACT_TEMPLATE
```

### Control Policy Configuration

The following example shows the configuration of a control policy that uses the template named LOW_IMPACT_TEMPLATE to provide limited access to all hosts even when authentication fails. If authentication succeeds, the policy manager removes the service template and provides access based on the policies downloaded by the RADIUS server.

```
class-map type control subscriber match-all DOT1X_MAB_FAILED
 no-match result-type method dot1x success
 no-match result-type method mab success
!
policy-map type control subscriber CONCURRENT_DOT1X_MAB_LOW_IMP_MODE
 event session-started match-all
  10 class always do-until-failure
   10 authorize
   20 activate service-template LOW_IMPACT_TEMPLATE
   30 authenticate using mab
   40 authenticate using dot1x
 event authentication-success match-all
  10 class always do-until-failure
   10 deactivate service-template LOW_IMPACT_TEMPLATE
 event authentication-failure match-first
  10 class DOT1X_MAB_FAILED do-until-failure
   10 authorize
   20 terminate dot1x
   30 terminate mab
 event agent-found match-all
  10 class always do-until-failure
   10 authenticate using dot1x
 event inactivity-timeout match-all
  10 class always do-until-failure
   10 clear-session
```

# Feature History for Identity Service Templates

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

*Table 38: Feature History for Identity Service Templates*

| Release | Feature Name | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Identity Service Template | Enables identity service templates to be configured locally and available at all times.<br>Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches |
| Cisco IOS XE Fuji 16.8.1a | Identity Service Template | Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 36

# Interface Templates

An interface template provides a mechanism to configure multiple commands at the same time and associate it with a target such as an interface. An interface template is a container of configurations or policies that can be applied to specific ports.


- Restrictions for Interface Templates, on page 673
- Information About Interface Templates, on page 673
- How to Configure Interface Templates, on page 677
- Configuration Examples for Interface Templates, on page 686
- Feature History for Interface Templates, on page 687


# Restrictions for Interface Templates

- Interface templates are not applicable for wireless sessions.
- Remote storing and downloading of templates is not supported.
- The same configuration cannot be used for port and interface template on the switch.

# Information About Interface Templates

## About Interface Templates

An interface template is a container of configurations or policies that can be applied to specific ports. When an interface template is applied to an access port, it impacts all traffic that is exchanged on the port.

There are two types of interface templates; user and builtin templates. Builtin templates are created by the system.

You can modify builtin templates. If you delete a modified builtin template the system restores the original definition of the template.

The following are the available builtin templates:

- AP_INTERFACE_TEMPLATE (Access Point)
- DMP_INTERFACE_TEMPLATE (Digital Media Player)

- IP_CAMERA_INTERFACE_TEMPLATE
- IP_PHONE_INTERFACE_TEMPLATE
- LAP_INTERFACE_TEMPLATE (Lightweight Access Point)
- MSP_CAMERA_INTERFACE_TEMPLATE
- MSP_VC_INTERFACE_TEMPLATE (Video Conferencing)
- PRINTER_INTERFACE_TEMPLATE
- ROUTER_INTERFACE_TEMPLATE
- SWITCH_INTERFACE_TEMPLATE
- TP_INTERFACE_TEMPLATE (TelePresence)

Following is an example of a builtin interface template:

```
Template Name       : IP_CAMERA_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security
 mls qos trust dscp
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
```

You can also create specific user templates with the commands that you want to include.

**Note** The template name must not contain spaces.

You can create an interface template using the **template** command in global configuration mode. In template configuration mode, enter the required commands. The following commands can be entered in template configuration mode:

| Command | Description |
|---|---|
| **access-session** | Configures access session specific interface commands. |
| **authentication** | Configures authentication manager Interface Configuration commands. |
| **carrier-delay** | Configures delay for interface transitions. |
| **dampening** | Enables event dampening. |
| **default** | Sets a command to its defaults. |
| **description** | Configures interface-specific description. |
| **dot1x** | Configures interface configuration commands for IEEE 802.1X. |

| Command | Description |
|---|---|
| **hold-queue** | Sets hold queue depth. |
| **ip** | Configures IP template. |
| **keepalive** | Enables keepalive. |
| **load-interval** | Specifies interval for load calculation for an interface. |
| **mab** | Configures MAC authentication bypass Interface. |
| **mls** | Enables multilayer switching configurations. This command is available on the following devices in template configuration mode:<br>• Cisco Catalyst 2960-S Series Switches<br>• Cisco Catalyst 2960-X Series Switches<br>• Cisco Industrial Ethernet 3000 Series Switches |
| **peer** | Configures peer parameters for point to point interfaces. |
| **priority-queue** | To set the priority-queue size for a template. This command is available on the following devices in template configuration mode:<br>• Cisco Catalyst 2960-S Series Switches<br>• Cisco Catalyst 2960-X Series Switches<br>• Cisco Industrial Ethernet 3000 Series Switches |
| **queue-set** | Configures the QoS queue set on a template. This command is available on the following devices in template configuration mode:<br>• Cisco Catalyst 2960-S Series Switches<br>• Cisco Catalyst 2960-X Series Switches<br>• Cisco Industrial Ethernet 3000 Series Switches |
| **radius-server** | Enables RADIUS server configurations. This command is available on the following devices in template configuration mode:<br>• Catalyst 4500E Supervisor Engine 7-E<br>• Catalyst 4500E Supervisor Engine 7L-E<br>• Catalyst 4500E Supervisor Engine 8-E<br>• Catalyst 4500-X Series Switches |
| **service-policy** | Configures CPL service policy. |

| Command | Description |
|---|---|
| **source** | Gets configurations from another source. |
| **spanning-tree** | Configures spanning tree subsystem |
| **storm-control** | Configures storm control. |
| **subscriber** | Configures subscriber inactivity timeout value. |
| **switchport** | Sets switching mode configurations |
| **trust** | Sets trust value for the interface. |

**Note**

- System builtin templates are not displayed in the running configuration. These templates show up in the running configuration only if you edit them.
- The stateful switchover fails if **access-session** and **swithcport mode access** are both configured in an interface template. To avoid the switchover failure, configure the **switchport mode access** command on the interface, instead of in an interface template.
- When you configure an interface template, it is recommended that you enter all the required dependent commands on the same template. It is not recommended to configure the dependent commands on two different templates.

# Binding an Interface Template to a Target

Each template can be bound to a target. Template binding or sourcing can be either static or dynamic. Static binding of a template involves binding the template to a target, like an interface. Only one template can be bound at a time using static binding. Static binding of another template to the same target will unbind the previously bound template. To configure static binding, use the **source template** command in interface configuration mode.

Any number of templates can be bound dynamically to a target. To configure dynamic binding using builtin policy maps and parameter maps, enable the autoconf feature using the **autoconf enable** command.

**Note** You can have statically and dynamically bind templates on the same interface at a time.

# Priority for Configurations Using Interface Templates

Configuration applied through dynamically-bound templates has the highest priority, followed by configuration applied directly on the interface, and then configuration applied through statically-bound templates. When similar commands are present at different priority levels, the one at the highest priority is applied. If a configuration at a higher priority level is not applied, then the configuration with the next highest priority is applied to the target.

Multiple templates can be dynamically bound to a target. When multiple templates are dynamically bound, the template that is applied last has the highest priority.

To delete a template, you must remove the binding to all targets. If you bind a template that does not exist, a new template is created with no configurations.

# How to Configure Interface Templates

## Configuring Interface Templates

Perform the following task to create user interface templates:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **template** *name*<br>**Example:**<br>`Device(config)# template user-template1` | Creates a user template and enters template configuration mode.<br>**Note**<br>Builtin template are system-generated. |
| **Step 4** | **load-interval** *interval*<br>**Example:**<br>`Device(config-template)# load-interval 60` | Configures the sampling interval for statistics collections on the template.<br>**Note**<br>Builtin template are system-generated. |
| **Step 5** | **description** *description*<br>**Example:**<br>`Device(config-template)# description This is a user template` | Configures the description for the template. |
| **Step 6** | **keepalive** *number*<br>**Example:**<br>`Device(config-template)# Keepalive 60` | Configures the keepalive timer. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring Static Binding for Interface Templates

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface GigabitEthernet 1/0/12` | Specifies the interface type and number and enters interface configuration mode. |
| **Step 4** | **source template** *name*<br>**Example:**<br>`Device(config-if)# source template user-template1` | Statically applies an interface template to a target. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

### Example

To verify static binding use the **show running-config interface** *int-name* and the **show derived-config interface** *int-name* commands.

```
Device# show running-config interface GigabitEthernet 1/0/12

Building configuration...
```

```
Current configuration : 71 bytes
!
interface GigabitEthernet1/0/12
source template user-template1
end
```

```
Device# show derived-config interface GigabitEthernet 1/0/12
Building configuration...

Derived configuration : 108 bytes
!
interface GigabitEthernet1/0/12
description This is a user template
load-interval 60
keepalive 60
end
```

# Configuring Dynamic Binding of Interface Templates

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *type number*<br>**Example:**<br>`Device(config)# interface GigabitEthernet 4/0/1` | Specifies the interface type and number and enters interface configuration mode. |
| **Step 4** | **service-policy type control subscriber** *policymap-name*<br>**Example:**<br>`Device(config-if)# service-policy type control subscriber POLICY-Gi1/0/12` | Dynamically applies an interface template to a target. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

# Verifying an Interface Template

Use one or more of the commands listed below to verify the interface template configuration.

## Procedure

**Step 1** **enable**

**Example:**

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

**Step 2** **show template interface all** {**all** | **binding** {*temp-name* | **all** | **target** *int-name*} | **brief** }
Shows all interface template configurations.

**Step 3** **show template interface source** {**built-in** [**original**] | **user**}{*temp-name* | **all**}}
Shows interface template source configurations.

**Step 4** **show template service**{**all** | **binding target** *int-name* | **brief** | **source** {**aaa** | **built-in** | **user** {**temp-name** | **all**}}
Shows all interface template service configurations.

### Verifying Interface User Templates

### Verifying all Builtin Templates

### Verifying all Builtin Templates on Cisco Catalyst 2960-S Series Switches , Cisco Catalyst 2960-X Series Switches, Cisco Industrial Ethernet 3000 Series Switches

### Verifying all Interface Templates Binding for all templates

### Verifying Static Template Binding for a Target Interface

### Verifying Dynamic Template Binding for all templates

### Verifying Template Binding for a Target Interface

```
Device# show template interface source user all
    Template Name : TEST-1
    Template Definition:
    load-interval 60
    description TEST_1_TEMPLATE
    keepalive 200
    !
    Template Name : TEST-2
    Template Definition:
    load-interval 60
```

```
    description TEST-1_TEMPLATE
    keepalive 200

Device#  show template interface source built-in all

Building configuration...

Template Name : AP_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode trunk
switchport nonegotiate
service-policy input AutoConf-4.0-Trust-Cos-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
!
Template Name : DMP_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport block unicast
switchport port-security
spanning-tree portfast
spanning-tree bpduguard enable
service-policy input AutoConf-4.0-Trust-Dscp-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
!
Template Name : IP_CAMERA_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport block unicast
switchport port-security
spanning-tree portfast
spanning-tree bpduguard enable
service-policy input AutoConf-4.0-Trust-Dscp-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
!
Template Name : IP_PHONE_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport block unicast
switchport port-security maximum 3
switchport port-security maximum 2 vlan access
switchport port-security violation restrict
switchport port-security aging time 2
switchport port-security aging type inactivity
switchport port-security
storm-control broadcast level pps 1k
storm-control multicast level pps 2k
storm-control action trap
spanning-tree portfast
spanning-tree bpduguard enable
service-policy input AutoConf-4.0-CiscoPhone-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
ip dhcp snooping limit rate 15
load-interval 30
!
Template Name : LAP_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport block unicast
switchport port-security violation protect
switchport port-security aging time 2
```

```
switchport port-security aging type inactivity
switchport port-security
storm-control broadcast level pps 1k
storm-control multicast level pps 2k
storm-control action trap
spanning-tree portfast
spanning-tree bpduguard enable
ip dhcp snooping limit rate 15
load-interval 30
!
Template Name : MSP_CAMERA_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport block unicast
switchport port-security
spanning-tree portfast
spanning-tree bpduguard enable
!
Template Name : MSP_VC_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport port-security
spanning-tree portfast
spanning-tree bpduguard enable
load-interval 30
!
Template Name : PRINTER_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport port-security maximum 2
switchport port-security
spanning-tree portfast
spanning-tree bpduguard enable
load-interval 60
!
Template Name : ROUTER_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode trunk
spanning-tree portfast trunk
spanning-tree bpduguard enable
service-policy input AutoConf-4.0-Trust-Cos-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
!
Template Name : SWITCH_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode trunk
service-policy input AutoConf-4.0-Trust-Cos-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
!
Template Name : TP_INTERFACE_TEMPLATE
Modified : No
Template Definition :
switchport mode access
switchport port-security maximum 3
switchport port-security maximum 2 vlan access
switchport port-security violation restrict
switchport port-security aging time 2
switchport port-security aging type inactivity
switchport port-security
```

```
storm-control broadcast level pps 1k
storm-control multicast level pps 2k
storm-control action trap
spanning-tree portfast
spanning-tree bpduguard enable
service-policy input AutoConf-4.0-Trust-Dscp-Input-Policy
service-policy output AutoConf-4.0-Output-Policy
ip dhcp snooping limit rate 15
load-interval 30
!
end
```

```
Device# show template interface source built-in all

Building configuration...

Template Name       : AP_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 switchport mode trunk
 switchport nonegotiate
 mls qos trust cos
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : DMP_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security
 mls qos trust dscp
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : IP_CAMERA_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security
 mls qos trust dscp
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : IP_PHONE_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security maximum 3
 switchport port-security maximum 2 vlan access
 switchport port-security violation  restrict
 switchport port-security aging time 2
 switchport port-security aging type inactivity
 switchport port-security
 storm-control broadcast level pps 1k
 storm-control multicast level pps 2k
 storm-control action trap
```

```
 mls qos trust cos
 service-policy input AUTOCONF-SRND4-CISCOPHONE-POLICY
 ip dhcp snooping limit rate 15
 load-interval 30
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : LAP_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security violation  protect
 switchport port-security aging time 2
 switchport port-security aging type inactivity
 switchport port-security
 storm-control broadcast level pps 1k
 storm-control multicast level pps 2k
 storm-control action trap
 mls qos trust dscp
 ip dhcp snooping limit rate 15
 load-interval 30
 srr-queue bandwidth share 10 10 60 20
 priority-queue out
!
Template Name       : MSP_CAMERA_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security
!
Template Name       : MSP_VC_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security violation  restrict
 switchport port-security aging time 2
 switchport port-security aging type inactivity
 switchport port-security
 ip dhcp snooping limit rate 15
 load-interval 30
!
Template Name       : PRINTER_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport port-security maximum 2
 switchport port-security
 load-interval 60
!
Template Name       : ROUTER_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast trunk
```

```
 spanning-tree bpduguard enable
 switchport mode trunk
 mls qos trust dscp
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : SWITCH_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 switchport mode trunk
 mls qos trust cos
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : TP_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport port-security maximum 3
 switchport port-security maximum 2 vlan access
 switchport port-security violation  restrict
 switchport port-security aging time 2
 switchport port-security aging type inactivity
 switchport port-security
 storm-control broadcast level pps 1k
 storm-control multicast level pps 2k
 storm-control action trap
 ip dhcp snooping limit rate 15
 load-interval 30
!
End
```

```
Device# show template interface binding all
    Template-name               Source      Method          Interface
    =============               ====      ==========      ==========
IP_PHONE_INTERFACE_TEMPLATE    Built-in    Dynamic        Gi1/0/1, Gi1/0/2, Gi1/0/3
                                                          Gi1/0/4, Gi1/0/5, Gi1/0/6
                                                          Gi1/0/7, Gi1/0/8, Gi1/0/9
                                                          Gi1/0/10, Gi1/0/11, Gi1/0/12
                                                          Gi1/0/13, Gi1/0/14, Gi1/0/15
                                                          Gi1/0/16, Gi1/0/17, Gi1/0/18
                                                          Gi1/0/19, Gi1/0/20, Gi1/0/21
                                                          Gi1/0/22, Gi1/0/23, Gi1/0/24
                                                          Gi1/1/1, Gi1/1/2, Gi1/1/3

IP_PHONE_INTERFACE_TEMPLATE    Built-in     Static        Gi4/0/4
```

```
Device# show template interface binding target GigabitEthernet 1/0/4
    Interface            Method       Source            Template
    =========           ==========    =====             =========
    Gi1/0/4              Dynamic      built-in          IP_PHONE_INTERFACE_TEMPLATE
                         Static       user              TEST
                         Dynamic      Modified-built-in TEST
```

```
Device# show template service all

    User-defined template:
    ======================

    Template Name      : SVC-1
    Template Definition:
    vlan 100
    access-group acl1
```

```
    built-in template:
    ======================

    Template Name      : SVC-2
    Template Definition:
    vlan 100
    access-group acl1

    aaa downloaded template:
    ========================
    Template Name      : SVC-2
    Template Definition:
    vlan 100
    access-group acl1

Device# show template binding target GigabitEthernet 1/0/4

  Interface Templates:
    Interface            method       Source            Template
    =========           ==========    =====             =========
    Gi1/0/4              Dynamic      built-in          IP_PHONE_INTERFACE_TEMPLATE
                         Static       user              TEST
                         Dynamic      Modified-built-in TEST
  Service Templates:
   Template            Source         Session-Mac
   ========            ====           ================
   SVC1                user           aa-bb-cc-dd-ee-ff
   SVC2                built-in       ab-ab-ab-ab-ab-ab
   SVC3                aaa            ac-ac-ac-ac-ac-ac
```

# Configuration Examples for Interface Templates

## Example: Configuring User Interface Templates

### Example: Configuring User Templates

```
Device# enable
Device (config)# configure terminal
Device(config)# template user-template1
Device(config-template)# load-interval 60
Device(config-template)# description This is a user template
Device(config-template)# Keepalive 60
Device(config)# end
```

## Example: Sourcing Interface Templates

```
Device> enable
Device# configure terminal
Device(config)# interface fastethernet 4/0/0
Device(config-if)# source template user-template1
Device(config-if)# end
```

## Example: Dynamically Binding Interface Templates

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 4/0/1
Device(config-if)# service-policy type control subscriber POLICY_Gi1/0/12
Device(config-if)# end
```

# Feature History for Interface Templates

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

***Table 39: Feature History for Interface Templates***

| Release | Feature Name | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Interface Templates | An interface template provides a mechanism to configure multiple commands at the same time and associate it with a target such as an interface.<br>Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches |
| Cisco IOS XE Fuji 16.8.1a | Interface Templates | Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 37

# Autoconf

Autoconf is a solution that can be used to manage port configurations for data or voice VLAN, quality of service (QoS) parameters, storm control, and MAC-based port security on end devices that are deployed in the access layer of a network.


- Prerequisites for Autoconf, on page 689
- Restrictions for Autoconf, on page 689
- Information About Autoconf, on page 690
- How to Configure Autoconf, on page 695
- Configuration Examples for Autoconf, on page 704
- Feature History for Autoconf, on page 705


## Prerequisites for Autoconf

- Before enabling Autoconf, disable the Auto SmartPort (ASP) macro, device classifier, and then access the session monitor.

## Restrictions for Autoconf

- Interface templates are not applicable for wireless sessions.
- When the Autoconf feature is enabled using the **autoconf enable** command, the default Autoconf service policy is applied to all interfaces. No other service policy can be applied globally using the **service-policy** command. To apply a different service policy, you must disable Autoconf on that interface. When a service policy is applied globally, you must disable it before enabling the Autoconf feature.
- When both local (interface-level) and global service policies exist, the local policy take precedence. Events in the local service policy are handled and the global service policy is not applied. The global service policy comes into effect only when the local policy is removed.
- Service templates cannot be applied to interfaces, and interface templates cannot be applied to service instances.
- Only one service template can be nested inside an interface template.

# Information About Autoconf

## Benefits of Autoconf

The Autoconf feature permits hardbinding between the end device and the interface. Autoconf falls under the umbrella of the Smart Operations solution. Smart Operations is a comprehensive set of capabilities that can simplify and improve LAN switch deployment. Smart Operations help organizations deliver operational excellence and scale services on the network.

The Autoconf feature automatically applies the needed configurations on the device ports to enable the efficient performance of each directly connected end device using a set of interface configurations that are configured inside an interface template.

- Autoconf efficiently applies commands to an interface because the parser does not need to parse each command each time.
- Configurations that are applied through the Autoconf feature can be reliably removed from a port without impacting previous or subsequent configurations on the port.
- The Autoconf feature provides built-in and user-defined configurations using interface and service templates. Configurations applied through templates can be centrally updated with a single operation.
- Using the Autoconf feature, a configuration can be applied to ports and access sessions.
- The Autoconf feature reduces ongoing maintenance for devices and attached end devices by making them intuitive and autoconfigurable. This reduces operation expenses (OPEX) and lowers the total cost of ownership (TCO).

## Identity Session Management and Templates

A key advantage of the Autoconf feature is that the core session management capability is decoupled from the application-specific logic; thus, allowing the same framework to be used regardless of the criteria for policy determination or the nature of the policies applied.

The identity session management infrastructure allows configurations and/or policies to be applied as templates.

Both service and interface templates are named containers of configuration and policy. Service templates may be applied only to access sessions, while interface templates may be applied only to ports. When a service template is applied to an access session, the contained configuration/policy is applied only to the target session and has no impact on other sessions that may be hosted on the same access port. Similarly, when an interface template is applied to an access port, it impacts all traffic exchanged on the port.

The Autoconf feature uses a set of built-in maps and built-in templates. The built-in templates are designed based on best practices for interface configurations. Built-in templates can be modified by the user to include customized configurations, limiting the need to create a new template.

The templates created by users are referred to as user-defined templates. User-defined templates can be defined on the device and can be mapped to any built-in or user-defined trigger.

Use the **show derived-config** command, to view the overall applied configurations applied by Autoconf template and manual configuration. The interface commands shown in the output of **show running-config interface** *type number* command are not necessarily the operational configuration. The Autoconf feature

dynamically applies a template to the interface, and overrides any conflicting static configuration that is already applied.

# Autoconf Operation

Autoconf uses the Device Classifier to identify the end devices that are connected to a port.

The Autoconf feature uses the device classification information gleaned from Cisco Discovery Protocol, LLDP, DHCP, MAC addresses, and the Organizationally Unique Identifier (OUI) that is identified by the Device Classifier.

The Device Classifier provides improved device classification capabilities and accuracy, and increased device visibility for enhanced configuration management.

Device classification is enabled when you enable the Autoconf feature using **autoconf enable** command in global configuration mode.

The device detection acts as an event trigger, which in turn applies the appropriate automatic template to the interface.

The Autoconf feature is based on a three-tier hierarchy.

- A policy map identifies the trigger type for applying the Autoconf feature.
- A parameter map identifies the appropriate template that must be applied, based on the end device.
- The templates contain the configurations to be applied.

The Autoconf built-in templates and triggers perform the these three steps automatically.

The Autoconf feature provides the following built-in templates:

- AP_INTERFACE_TEMPLATE
- DMP_INTERFACE_TEMPLATE
- IP_CAMERA_INTERFACE_TEMPLATE
- IP_PHONE_INTERFACE_TEMPLATE
- LAP_INTERFACE_TEMPLATE
- MSP_CAMERA_INTERFACE_TEMPLATE
- MSP_VC_INTERFACE_TEMPLATE
- PRINTER_INTERFACE_TEMPLATE
- ROUTER_INTERFACE_TEMPLATE
- SWITCH_INTERFACE_TEMPLATE
- TP_INTERFACE_TEMPLATE

**Note** By default built-in templates are not displayed under running configuration. The built-in templates show in the running configuration only if you edit them.

The template that is selected is based on parameter map information applied to an interface. This information can be based on the following criteria:

- End Device type
- MAC address
- OUI
- User role
- Username

The Autoconf feature provides one built-in parameter map BUILTIN_DEVICE_TO_TEMPLATE with the following configuration:

```
Parameter-map name: BUILTIN_DEVICE_TO_TEMPLATE
 Map: 10 map device-type regex "Cisco-IP-Phone"
  Action(s):
   20 interface-template IP_PHONE_INTERFACE_TEMPLATE
 Map: 20 map device-type regex "Cisco-IP-Camera"
  Action(s):
   20 interface-template IP_CAMERA_INTERFACE_TEMPLATE
 Map: 30 map device-type regex "Cisco-DMP"
  Action(s):
   20 interface-template DMP_INTERFACE_TEMPLATE
 Map: 40 map oui eq "00.0f.44"
  Action(s):
   20 interface-template DMP_INTERFACE_TEMPLATE
 Map: 50 map oui eq "00.23.ac"
  Action(s):
   20 interface-template DMP_INTERFACE_TEMPLATE
 Map: 60 map device-type regex "Cisco-AIR-AP"
  Action(s):
   20 interface-template AP_INTERFACE_TEMPLATE
 Map: 70 map device-type regex "Cisco-AIR-LAP"
  Action(s):
   20 interface-template LAP_INTERFACE_TEMPLATE
 Map: 80 map device-type regex "Cisco-TelePresence"
  Action(s):
   20 interface-template TP_INTERFACE_TEMPLATE
 Map: 90 map device-type regex "Surveillance-Camera"
  Action(s):
   10 interface-template MSP_CAMERA_INTERFACE_TEMPLATE
 Map: 100 map device-type regex "Video-Conference"
  Action(s):
   10 interface-template MSP_VC_INTERFACE_TEMPLATE
```

**Note** Use the **show parameter-map type subscriber attribute-to-service All** command to view the configuration for the built-in parameter map.

The Autoconf feature provides one built-in policy map BUILTIN_AUTOCONF_POLICY with the following configuration:

```
BUILTIN_AUTOCONF_POLICY
  event identity-update match-all
    10 class always do-until-failure
      10 map attribute-to-service table BUILTIN_DEVICE_TO_TEMPLATE
```

**Note** Use the **show policy-map type control subscriber BUILTIN_AUTOCONF_POLICY** command to view the configuration for the built-in policy map.

You can also manually create policy maps, parameter maps, and templates.

When a trigger is created that is based on specific user information, a local 802.1X Cisco Identity Services Engine (ISE) server authenticates it ensuring the security of the operation.

An interface template can be dynamically activated (on an interface) using any of the following methods:

- RADIUS CoA—While Change of Authorization (CoA) commands are targeted to one or more access sessions, any referenced template must be applied to the interface hosting the referenced session.
- RADIUS Access-Accept for client authentication or authorization—Any referenced interface template returned in an Access-Accept must be applied to the port that is hosting the authorized access session.
- Service template—If an interface template is referenced in a service template that is either locally defined or sourced from the AAA server, the interface template must be applied to the interface hosting any access-session on which the service template is applied (add a new command for interface template reference from within a locally defined service template).
- Subscriber control-policy action—A mapping action under the subscriber control policy activates service and/or interface template (as referenced in a parameter map) based on the type of filter, and removes any templates associated with a previous policy.
- Device-to-template parameter map—A subscriber parameter map that allows the filter type to service and/or interface template mappings to be specified in an efficient and readable manner.

# Advantages of Using Templates

Using templates for autoconfiguration has the following benefits:

- Templates are parsed once when they are being defined. This makes dynamic application of the templates very efficient.
- Templates can be applied to an Ethernet interface that is connected to an end device, based on the type of the end device.
- Service templates allow the activation of session-oriented features, whereas interface templates apply configurations to the interface that is hosting a session.
- Service templates are applied to access sessions and hence only impact the traffic exchanged with a single endpoint on a port.
- Startup and running configurations of the device are not modified by the dynamic application of the template.
- Policy application is synchronized with the access-session life cycle, which is tracked by the framework by using all available techniques, including just link-up/link-down.
- Templates can be updated with a single operation. All applied instances of the templates are updated.
- Constituent commands of the templates do not appear in the running configuration.

- Templates can be removed with no impact on previous or subsequent configurations.
- Template application is acknowledged, allowing for synchronization and performing remedial actions where failures occur.
- Data VLAN, quality of service (QoS) parameters, storm control, and MAC-based port security are configured automatically based on the end device that is connected to the switch.
- The switch port is cleaned up completely by removing configurations when the device is disconnected from a port.
- Human error is reduced in the installation and configuration process.

# Autoconf Functionality

The Autoconf feature is disabled by default in global configuration mode. When you enable the Autoconf feature in global configuration mode, it is enabled by default at the interface level. The built-in template configurations are applied based on the end devices detected on all interfaces.

Use the **access-session inherit disable autoconf** command to manually disable Autoconf at the interface level, even when Autoconf is enabled at the global level.

If you disable Autoconf at the global level, all interface-level configurations are disabled.

| Global | Interface Level | AutoConf Status |
|---|---|---|
| Disable | Disable | No automatic configurations are applied when an end device is connected. |
| Enable | Enabled by default | If Autoconf is enabled at the global level, it is enabled at the interface level by default. Built-in template configurations are applied based on the end devices that are detected on all interfaces. |
| Enable | Disable | Enabled at global level. Disabled at interface level. No automatic configurations are applied when an end device is connected to the interface on which Autoconf is disabled. |

Autoconf allows you to retain the template even when the link to the end device is down or the end device is disconnected, by configuring the Autoconf sticky feature. Use the **access-session interface-template sticky** command to configure the Autoconf sticky feature in global configuration mode. The Autoconf sticky feature avoids the need for detecting the end device and applying the template every time the link flaps or device is removed and connected back.

The **access-session interface-template sticky** command is mandatory to apply an inbuilt template that contains **access-session** commands on an interface. Configure the **access-session interface-template sticky** command to apply interface template on a port using a service policy.

If you want to disable the Autoconf feature on a specific interface, use the **access-session inherit disable interface-template-sticky** command in interface configuration mode.

# How to Configure Autoconf

## Applying a Built-in Template to an End Device

The following task shows how to apply a built-in template on an interface that is connected to an end device, for example, a Cisco IP phone.

### Before you begin

Make sure that the end device, for example, a Cisco IP phone, is connected to the switch port.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device(config)# configure terminal` | Enters global configuration mode. |
| **Step 3** | **autoconf enable**<br>**Example:**<br>`Device(config)# autoconf enable` | Enables the Autoconf feature. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and enters privileged EXEC mode. |
| **Step 5** | (Optional) **show device classifier attached interface** *interface-type* *interface-number*<br>**Example:**<br>`Device# show device classifier attached interface Gi3/0/26` | Displays whether the end device is classified by the device classifier with correct attributes. |
| **Step 6** | **show template binding target** *interface-type interface-number*<br>**Example:**<br>`Device# show template binding target gi3/0/26` | Displays the configuration applied through the template on the interface. |

### Verifying th device classification of an End Device

### Verifying the Interface Template on an Interface

### Verifying the Interface Configuration

### Verifying Interface Configuration for Cisco IOS 4500 Series, Cisco IOS 3650 Series, Cisco IOS 3560 Series, and Cisco IOS 2960 Switches

### Verifying Global Configuration after Applying Autoconf

The following example shows that an IP phone is classified by the Device Classifier with correct attributes:

```
Device# show device classifier attached interface GigabitEthernet 3/0/26

Summary:

MAC_Address       Port_Id       Profile Name                        Device Name
==============    ==========    ==================                  ===================
0026.0bd9.7bbb    Gi3/0/26      Cisco-IP-Phone-7962                 Cisco IP Phone 7962
```

The following example shows that a built-in interface template is applied on the interface:

```
Device# show template binding target GigabitEthernet 3/0/26

Interface Templates
===================
Interface: Gi4/0/11
Method              Source            Template-Name
------              ------            -------------
dynamic             Built-in          IP_PHONE_INTERFACE_TEMPLATE
```

The following example shows how to verify the interface configuration after the interface template is applied to the IP phone connected to the GigabitEthernet interface 3/0/26 :

```
Device# show running-config interface GigabitEthernet 3/0/26
Building configuration...

Current configuration : 624 bytes
!
interface GigabitEthernet3/0/26
!
End

Device# show derived-config interface GigabitEthernet 3/0/26

Building configuration...

Derived configuration : 649 bytes
!
interface GigabitEthernet3/0/26
 switchport mode access
 switchport block unicast
 switchport port-security maximum 3
 switchport port-security maximum 2 vlan access
 switchport port-security violation restrict
 switchport port-security aging time 2
 switchport port-security aging type inactivity
 switchport port-security
```

```
 load-interval 30
 storm-control broadcast level pps 1k
 storm-control multicast level pps 2k
 storm-control action trap
 spanning-tree portfast
 spanning-tree bpduguard enable
 service-policy input AutoConf-4.0-CiscoPhone-Input-Policy
 service-policy output AutoConf-4.0-Output-Policy
 ip dhcp snooping limit rate 15
end
```

The following example shows how to verify the interface configuration:

```
Device# show template interface source built-in all
Building configuration...

Template Name       : AP_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 switchport mode trunk
 switchport nonegotiate
 mls qos trust cos
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : DMP_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security
 mls qos trust dscp
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : IP_CAMERA_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security
 mls qos trust dscp
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : IP_PHONE_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security maximum 3
 switchport port-security maximum 2 vlan access
 switchport port-security violation  restrict
 switchport port-security aging time 2
 switchport port-security aging type inactivity
 switchport port-security
 storm-control broadcast level pps 1k
 storm-control multicast level pps 2k
 storm-control action trap
```

```
 mls qos trust cos
 service-policy input AUTOCONF-SRND4-CISCOPHONE-POLICY
 ip dhcp snooping limit rate 15
 load-interval 30
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : LAP_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security violation  protect
 switchport port-security aging time 2
 switchport port-security aging type inactivity
 switchport port-security
 storm-control broadcast level pps 1k
 storm-control multicast level pps 2k
 storm-control action trap
 mls qos trust dscp
 ip dhcp snooping limit rate 15
 load-interval 30
 srr-queue bandwidth share 10 10 60 20
 priority-queue out
!
Template Name       : MSP_CAMERA_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security
!
Template Name       : MSP_VC_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport block unicast
 switchport port-security violation  restrict
 switchport port-security aging time 2
 switchport port-security aging type inactivity
 switchport port-security
 ip dhcp snooping limit rate 15
 load-interval 30
!
Template Name       : PRINTER_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport port-security maximum 2
 switchport port-security
 load-interval 60
!
Template Name       : ROUTER_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast trunk
```

```
 spanning-tree bpduguard enable
 switchport mode trunk
 mls qos trust dscp
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : SWITCH_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 switchport mode trunk
 mls qos trust cos
 srr-queue bandwidth share 1 30 35 5
 priority-queue out
!
Template Name       : TP_INTERFACE_TEMPLATE
Modified            : No
Template Definition :
 spanning-tree portfast
 spanning-tree bpduguard enable
 switchport mode access
 switchport port-security maximum 3
 switchport port-security maximum 2 vlan access
 switchport port-security violation  restrict
 switchport port-security aging time 2
 switchport port-security aging type inactivity
 switchport port-security
 storm-control broadcast level pps 1k
 storm-control multicast level pps 2k
 storm-control action trap
 ip dhcp snooping limit rate 15
 load-interval 30
!
```

```
Device# show running config
class-map match-any AutoConf-4.0-Scavenger-Queue
 match dscp cs1
 match cos  1
 match access-group name AutoConf-4.0-ACL-Scavenger
class-map match-any AutoConf-4.0-VoIP
 match dscp ef
 match cos  5
class-map match-any AutoConf-4.0-Control-Mgmt-Queue
 match cos  3
 match dscp cs7
 match dscp cs6
 match dscp cs3
 match dscp cs2
 match access-group name AutoConf-4.0-ACL-Signaling
class-map match-any AutoConf-4.0-Multimedia-Conf
 match dscp af41
 match dscp af42
 match dscp af43
class-map match-all AutoConf-4.0-Broadcast-Vid
 match dscp cs5
class-map match-any AutoConf-4.0-Bulk-Data
 match dscp af11
 match dscp af12
 match dscp af13
class-map match-all AutoConf-4.0-Realtime-Interact
 match dscp cs4
class-map match-any AutoConf-4.0-VoIP-Signal
 match dscp cs3
 match cos  3
class-map match-any AutoConf-4.0-Trans-Data-Queue
 match cos  2
```

```
 match dscp af21
 match dscp af22
 match dscp af23
 match access-group name AutoConf-4.0-ACL-Transactional-Data
class-map match-any AutoConf-4.0-VoIP-Data
 match dscp ef
 match cos  5
class-map match-any AutoConf-4.0-Multimedia-Stream
 match dscp af31
 match dscp af32
 match dscp af33
class-map match-all AutoConf-4.0-Internetwork-Ctrl
 match dscp cs6
class-map match-all AutoConf-4.0-VoIP-Signal-Cos
 match cos  3
class-map match-any AutoConf-4.0-Multimedia-Stream-Queue
 match dscp af31
 match dscp af32
 match dscp af33
class-map match-all AutoConf-4.0-Network-Mgmt
 match dscp cs2
class-map match-all AutoConf-4.0-VoIP-Data-Cos
 match cos  5
class-map match-any AutoConf-4.0-Priority-Queue
 match cos  5
 match dscp ef
 match dscp cs5
 match dscp cs4
class-map match-any AutoConf-4.0-Bulk-Data-Queue
 match cos  1
 match dscp af11
 match dscp af12
 match dscp af13
 match access-group name AutoConf-4.0-ACL-Bulk-Data
class-map match-any AutoConf-4.0-Transaction-Data
 match dscp af21
 match dscp af22
 match dscp af23
class-map match-any AutoConf-4.0-Multimedia-Conf-Queue
 match cos  4
 match dscp af41
 match dscp af42
 match dscp af43
 match access-group name AutoConf-4.0-ACL-Multimedia-Conf
class-map match-all AutoConf-4.0-Network-Ctrl
 match dscp cs7
class-map match-all AutoConf-4.0-Scavenger
 match dscp cs1
class-map match-any AutoConf-4.0-Signaling
 match dscp cs3
 match cos  3
!
!
policy-map AutoConf-4.0-Cisco-Phone-Input-Policy
 class AutoConf-4.0-VoIP-Data-Cos
  set dscp ef
  police cir 128000 bc 8000
   exceed-action set-dscp-transmit cs1
   exceed-action set-cos-transmit 1
 class AutoConf-4.0-VoIP-Signal-Cos
  set dscp cs3
  police cir 32000 bc 8000
   exceed-action set-dscp-transmit cs1
   exceed-action set-cos-transmit 1
```

```
 class class-default
  set dscp default
  set cos 0
policy-map AutoConf-4.0-Output-Policy
 class AutoConf-4.0-Scavenger-Queue
  bandwidth remaining percent 1
 class AutoConf-4.0-Priority-Queue
  priority
  police cir percent 30 bc 33 ms
 class AutoConf-4.0-Control-Mgmt-Queue
  bandwidth remaining percent 10
 class AutoConf-4.0-Multimedia-Conf-Queue
  bandwidth remaining percent 10
 class AutoConf-4.0-Multimedia-Stream-Queue
  bandwidth remaining percent 10
 class AutoConf-4.0-Trans-Data-Queue
  bandwidth remaining percent 10
    dbl
 class AutoConf-4.0-Bulk-Data-Queue
  bandwidth remaining percent 4
    dbl
 class class-default
  bandwidth remaining percent 25
    dbl
policy-map AutoConf-DMP
 class class-default
  set dscp cs2
policy-map AutoConf-IPVSC
 class class-default
  set cos dscp table AutoConf-DscpToCos
policy-map AutoConf-4.0-Input-Policy
 class AutoConf-4.0-VoIP
 class AutoConf-4.0-Broadcast-Vid
 class AutoConf-4.0-Realtime-Interact
 class AutoConf-4.0-Network-Ctrl
 class AutoConf-4.0-Internetwork-Ctrl
 class AutoConf-4.0-Signaling
 class AutoConf-4.0-Network-Mgmt
 class AutoConf-4.0-Multimedia-Conf
 class AutoConf-4.0-Multimedia-Stream
 class AutoConf-4.0-Transaction-Data
 class AutoConf-4.0-Bulk-Data
 class AutoConf-4.0-Scavenger
```

# Applying a Modified Built-in Template to an End Device

The following task shows how to modify a built-in template when multiple wireless access points and IP cameras are connected to a switch.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device(config)# configure terminal` | Enters global configuration mode. |
| **Step 3** | **template** *template-name*<br>**Example:**<br>`Device(config)# template AP_INTERFACE_TEMPLATE` | Enters template configuration mode for the builtin template. |
| **Step 4** | **switchport access vlan** *vlan-id*<br>**Example:**<br>`Device(config-template)# switchport access vlan 20` | Sets the VLAN when the interface is in access mode. |
| **Step 5** | **description** *description*<br>**Example:**<br>`Device(config-template)# description modifiedAP` | Modifies the description of the built-in template. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(config-template)# exit` | Exits template configuration mode and enters global configuration mode. |
| **Step 7** | **autoconf enable**<br>**Example:**<br>`Device(config)# autoconf enable` | Enables the Autoconf feature. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and enters privileged EXEC mode. |
| **Step 9** | **show template interface binding all**<br>**Example:**<br>`Device# show template interface binding all` | Displays whether the template is applied on the interface. |

### Verifying the Device classification of an End Device

### Verifying the Interface Template on an Interface

The following example shows that the IP camera and access points are classified by the Device Classifier with correct attributes:

```
Device# show device classifier attached detail

DC default profile file version supported = 1
```

```
Detail:
MAC_Address     Port_Id    Cert Parent Proto    ProfileType     Profile Name
Device_Name
==========================================================================================
001d.a1ef.23a8  Gi1/0/7     30    3    C     M    Default      Cisco-AIR-AP-1130      cisco
 AIR-AP1131AG-A-K9
001e.7a26.eb05  Gi1/0/30    70    2    C     M    Default      Cisco-IP-Camera        Cisco
 IP Camera
```

The following example shows that a built-in interface template is applied on the interface:

```
Device# show template interface binding all

Template-Name                   Source            Method          Interface
-------------                   ------            ------          ---------
IP_CAMERA_INTERFACE_TEMPLATE     Built-in          dynamic         Gi1/0/30
AP_INTERFACE_TEMPLATE          Modified-Built-in   dynamic         Gi1/0/7
```

# Migrating from ASP to Autoconf

### Before you begin

Verify that the AutoSmart Port (ASP) macro is running using the **show running-config | include macro auto global** command.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **no macro auto global processing**<br>**Example:**<br>`Device(config)# no macro auto global processing` | Disables ASP on a global level. |
| **Step 4** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and returns to privileged EXEC mode. |
| **Step 5** | **clear macro auto configuration all**<br>**Example:** | Clears macro configurations for all interfaces. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# clear macro auto configuration all` | |
| **Step 6** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 7** | **autoconf enable**<br>**Example:**<br>`Device(config)# autoconf enable` | Enables the Autoconf feature. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuration Examples for Autoconf

## Example: Applying a Built-in Template to an End Device

The following example shows how to apply a built-in template to an end device connected to an interface.

```
Device> enable
Device(config)# configure terminal
Device(config)# autoconf enable
Device(config)# end
Device# show device classifier attached interface Gi3/0/26
Device# show template binding target GigabitEthernet 3/0/26
```

## Example: Applying a Modified Built-in Template to an End Device

The following example shows how to modified built-in template and verify the configuration:

```
Device> enable
Device(config)# configure terminal
Device(config)# template AP_INTERFACE_TEMPLATE
Device(config-template)# switchport access vlan 20
Device(config-template)# description modifiedAP
Device(config-template)# exit
Device(config)# autoconf enable
Device(config)# end
Device# show template interface binding all
```

## Example: Migrating from ASP Macros to Autoconf

The following example shows how to migrate from ASP to Autoconf:

```
Device> enable
Device# configure terminal
Device(config)# no macro auto global processing
Device(config)# exit
Device# clear macro auto configuration all
Device# configure terminal
Device(config)# autoconf enable
Device(config)# end
```

# Feature History for Autoconf

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

***Table 40: Feature History for Autoconf***

| Release | Feature Name | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Autoconf | Autoconf is a solution that can be used to manage port configurations for data or voice VLANs, quality of QoS parameters, storm control, and MAC-based port security on end devices that are deployed in the access layer of a network.<br><br>The Autoconf feature automatically applies the configurations needed on the device ports to enable the efficient performance of each directly connected end device using a set of interface configurations that are configured inside an interface template. This mechanism ensures that no configurations are needed from the end device.<br><br>Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches |
| Cisco IOS XE Fuji 16.8.1a | Autoconf | Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 38

# eEdge Integration with MACsec

The Media Access Control Security (MACsec) standard is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. The eEdge Integration with MACsec feature allows you to integrate the MACsec standard with enterprise edge (eEdge) devices to enhance Session Aware Networking capabilities. Session Aware Networking provides a policy and identity-based framework for edge devices to deliver flexible and scalable services to subscribers.


- Prerequisites for eEdge Integration with MACsec, on page 707
- Restrictions for eEdge Integration with MACsec, on page 707
- Information About eEdge Integration with MACsec, on page 708
- How to Configure eEdge Integration with MACsec, on page 709
- Configuration Examples for eEdge Integration with MACsec, on page 711
- Feature History for eEdge Integration with MACsec, on page 712


## Prerequisites for eEdge Integration with MACsec

- Layer 2 encryption protocols like the IEEE 802.1AE Media Access Control Security (MACsec) standard must register with the eEdge session manager to receive disconnect notifications and perform cleanup.
- You must provision one virtual interface per secure association.

## Restrictions for eEdge Integration with MACsec

- The Media Access Control Security (MACsec) standard is supported only in single-host and multihost modes. If a link layer security policy is configured as must-secure and the host mode is not configured as a single host or a multihost, the connection is closed.
- The MACsec standard is not supported in multi-authentication mode.
- The MACsec standard supports the 802.1AE encryption with MACsec Key Agreement (MKA) only on downlink ports for encryption between a MACsec-capable device and host devices.

# Information About eEdge Integration with MACsec

The following sections provide information about eEdge Integration with MACsec feature.

## Overview of MACsec

Media Access Control Security (MACsec) is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. Implementing the MACsec encryption standard enables support for the 802.1AE encryption with MACsec Key Agreement (MKA) on downlink ports for encryption between a MACsec-capable device and host devices. The MACsec-capable device also supports MACsec link layer device-to-device security by using Cisco TrustSec Network Device Admission Control (NDAC) and the Security Association Protocol (SAP) key exchange. Link layer security includes both packet authentication between devices and MACsec encryption between devices (encryption is optional).

### MACsec Standard Encryption

The Media Access Control Security (MACsec) standard provides data link layer encryption over wired networks by using out-of-band methods for encryption keying. The MACsec Key Agreement (MKA) protocol provides the required session keys and manages the encryption keys. MKA and MACsec are implemented after a successful authentication by using the 802.1X Extensible Authentication Protocol (EAP) framework. Only host-facing links (links between network access devices and endpoint devices such as a PC or an IP phone) can be secured using MACsec.

A device that uses MACsec accepts either MACsec or non-MACsec frames, depending on the policy associated with the client. MACsec frames are encrypted and protected with an integrity check value (ICV). When the device receives frames from the client, it decrypts them and calculates the correct ICV by using session keys provided by MKA. The device compares the calculated value of the ICV to the ICV within the frame. If they are not identical, the frame is dropped. The device also encrypts and adds an ICV to any frame that is sent over a secured port (the access point used to provide the secure MAC service to a client) using the current session key.

The MKA protocol manages the encryption keys used by the underlying MACsec protocol. The basic requirements of MKA are defined in 802.1X-2010. The MKA protocol extends 802.1X to allow peer discovery with confirmation of mutual authentication and sharing of MACsec secret keys to protect data exchanged by peers.

### EAP Implementation of MKA

The Extensible Authentication Protocol (EAP) framework implements MKA as a newly defined EAP-over-LAN (EAPOL) packet. EAP authentication produces a master session key (MSK) that is shared by both partners in the data exchange. Entering the EAP session ID generates a secure connectivity association key name (CKN). Because the device is the authenticator, it is also the key server, generating a random 128-bit secure association key (SAK), which it sends it to the client partner. The client is never a key server and can only interact with a single MKA entity, the key server. After key derivation and generation, the device sends periodic transports to the partner at a default interval of 2 seconds.

The packet body in an EAPOL Protocol Data Unit (PDU) is referred to as a MACsec Key Agreement PDU (MKPDU). MKA sessions and participants are deleted when the MKA lifetime (6 seconds) passes and no MKPDU is received from a participant. For example, if a client disconnects, the participant on the device continues to operate MKA until 6 seconds have elapsed after the last MKPDU is received from the client.

# eEdge Integration with MACsec

The Media Access Control Security (MACsec) standard is the IEEE 802.1AE standard for authenticating and encrypting packets between two MACsec-capable devices. The eEdge Integration with MACsec feature allows you to integrate the MACsec standard with enterprise edge (eEdge) devices to enhance Session Aware Networking capabilities. Session Aware Networking provides a policy and identity-based framework for edge devices to deliver flexible and scalable services to subscribers.

# How to Configure eEdge Integration with MACsec

## Integrating eEdge with MACsec

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **service-template** *template-name*<br>**Example:**<br>`Device(config)# service-template dot1x-macsec-policy` | Defines a template that contains a set of service policy attributes to apply to subscriber sessions and enters service template configuration mode. |
| **Step 4** | **linksec policy** {**must-not-secure** \| **must-secure** \| **should-secure**}<br>**Example:**<br>`Device(config-service-template)# linksec policy must-secure` | Sets the link security policy as must-secure.<br>• Must-secure policy authorizes the eEdge device port only if a secure MACsec session is established. |
| **Step 5** | **exit**<br>**Example:**<br>`Device(config-service-template)# exit` | Exits service template configuration mode and returns to global configuration mode. |
| **Step 6** | **policy-map type control subscriber** *control-policy-name*<br>**Example:**<br>`Device(config)# policy-map type control subscriber cisco-subscriber` | Defines a control policy for subscriber sessions and enters control policy-map event configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | **event authentication-success** [**match-all** \| **match-any**]<br>**Example:**<br>`Device(config-event-control-policymap)# event authentication-success match-all` | Specifies the type of event that triggers actions in a control policy if all authentication events are a match and enters control policy-map class configuration mode. |
| **Step 8** | *priority-number* **class** {*control-class-name* \| **always**} [**do-all** \| **do-until-failure** \| **do-until-success**]<br>**Example:**<br>`Device(config-class-control-policymap)# 10 class always do-until-failure` | Specifies that the control class should execute the actions in a control policy, in the specified order, until one of the actions fails, and enters control policy-map action configuration mode. |
| **Step 9** | *action-number* **activate** {**policy type control subscriber** *control-policy-name* \| **service-template** *template-name* [**aaa-list** *list-name*] [**precedence** [**replace-all**]}<br>**Example:**<br>`Device(config-action-control-policymap)# 10 activate service-template dot1x-macsec-policy` | Activates a control policy on a subscriber session. |
| **Step 10** | **end**<br>**Example:**<br>`Device(config-action-control-policymap)# end` | Exits control policy-map action configuration mode and enters privileged EXEC mode. |

# Identifying Link Layer Security Failures

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 2** | **class-map type control subscriber** {**match-all** \| **match-any** \| **match-none**} *control-class-name*<br>**Example:**<br>`Device(config)# class-map type control subscriber match-all linksec-failed` | Creates a control class, which defines the conditions under which the actions of a control policy are executed and enters control class-map filter configuration mode. |
| **Step 3** | **match authorization-failure** {**domain-change-failed** \| **linksec-failed**} | Configures a match condition in a control class based on the type of authorization failure |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device(config-filter-control-classmap)# match authorization-failure linksec-failed` | received from an authorization failed event of a link layer security failure. |
| **Step 4** | **exit**<br>**Example:**<br>`Device(config-class-control-policymap)# exit` | Exits control class-map filter configuration mode and enters global configuration mode. |
| **Step 5** | **policy-map type control subscriber** *control-policy-name*<br>**Example:**<br>`Device(config)# policy-map type control subscriber cisco-subscriber` | Defines a control policy for subscriber sessions and enters control policy-map event configuration mode. |
| **Step 6** | **event authentication-failure** [**match-all** \| **match-any**]<br>**Example:**<br>`Device(config-event-control-policymap)# event authentication-failure match-all` | Specifies the type of event that triggers actions in a control policy if session authentication fails and enters control policy-map class configuration mode. |
| **Step 7** | *priority-number* **class** {*control-class-name* \| **always**} [**do-all** \| **do-until-failure** \| **do-until-success**]<br>**Example:**<br>`Device(config-class-control-policymap)# 10 class linksec-failed do-until-failure` | Specifies that the control class must execute the actions in a control policy, in the specified order, until one of the actions fails and enters control policy-map action configuration mode. |
| **Step 8** | **end**<br>**Example:**<br>`Device(config-action-control-policymap)# end` | Exits control policy-map action configuration mode and enters privileged EXEC mode. |

# Configuration Examples for eEdge Integration with MACsec

## Example: Integrating eEdge with MACsec

```
Device> enable
Device# configure terminal
Device(config)# service-template dot1x-macsec-policy
Device(config-service-template)# linksec policy must-secure
Device(config-service-template)# exit
Device(config)# policy-map type control subscriber cisco-subscriber
Device(config-event-control-policymap)# event authentication-success match-all
Device(config-class-control-policymap)# 10 class always do-until-failure
```

```
Device(config-action-control-policymap)# 10 activate service-template dot1x-macsec-policy
Device(config-action-control-policymap)# end
```

## Example: Identifying Linksec Failures

```
Device# configure terminal
Device(config)# class-map type control subscriber match-all linksec-failure
Device(config-filter-control-classmap)# match authorization-failure linksec-failed
Device(config-class-control-classmap)# exit
Device(config)# policy-map type control subscriber cisco-subscriber
Device(config-event-control-policymap)# event authentication-failure match-all
Device(config-class-control-policymap)# 10 class linksec-failed do-until-failure
Device(config-action-control-policymap)# end
```

# Feature History for eEdge Integration with MACsec

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

***Table 41: Feature History for eEdge Integration with MACsec***

| Release | Feature Name | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | eEdge Integration with MACsec | The eEdge Integration with MACsec feature allows you to integrate the MACsec standard with enterprise edge (eEdge) devices to enhance Session Aware Networking capabilities.<br><br>Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches |
| Cisco IOS XE Fuji 16.8.1a | eEdge Integration with MACsec | Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 39

# Critical Voice VLAN Support

The Critical Voice VLAN Support feature directs phone traffic to the configured voice VLAN of a port if the authentication server becomes unreachable.

With normal network connectivity, when an IP phone successfully authenticates on a port, the authentication server directs the phone traffic to the voice domain of the port. If the authentication server becomes unreachable, IP phones cannot authenticate the phone traffic. In multidomain authentication (MDA) mode or multiauthentication mode, you can configure the Critical Voice VLAN Support feature to direct phone traffic to the configured voice VLAN of the port. The phone is authorized as an unknown domain. Both data and voice are enabled for the phone.


- Restrictions for Critical Voice VLAN Support, on page 713
- Information About Critical Voice VLAN Support, on page 713
- How to Configure Critical Voice VLAN Support, on page 714
- Configuration Examples for Critical Voice VLAN Support, on page 719
- Feature History for Critical Voice VLAN Support, on page 719


# Restrictions for Critical Voice VLAN Support

- Different VLANs must be configured for voice and data.
- The voice VLAN must be configured on a device.
- The Critical Voice VLAN Support feature does not support standard Access Control Lists (ACLs) on the switch port.

# Information About Critical Voice VLAN Support

## Critical Voice VLAN Support in Multidomain Authentication Mode

If a critical voice VLAN is deployed using an interface in multidomain authentication (MDA) mode, the host mode is changed to multihost and the first phone device is installed as a static forwarding entries. Any additional phone devices are installed as dynamic forwarding entry in the Host Access Table (HAT).

**Note**

- If a critical port is already authorized and reauthentication occurs, the switch puts the port in the critical-authentication state in the current VLAN, which might be the one previously assigned by the RADIUS server.
- Inaccessible authentication bypass is compatible with guest VLAN. When a guest VLAN is enabled on a 802.1X port, the features interact as follows: if all RADIUS servers are not available and if a client is connected to a critical port and was previously assigned to a guest VLAN, the switch keeps the port in the guest VLAN.

## Critical Voice VLAN Support in Multiauthentication Mode

If the critical authentication feature is deployed in multiauthentication mode, only one phone device will be allowed and a second phone trying to authorize will trigger a violation.

The **show authentication sessions** command displays the critical voice client data. A critically authorized voice client in multiauthentication host mode will be in the “authz success” and “authz fail” state.

**Note** If critical voice is required, then critical data should be configured too. Otherwise, the critical voice client will be displayed in the “authz fail” state while the voice VLAN will be open.

## Critical Voice VLAN Support in a Service Template

On enterprise Edge (eEdge) devices, the critical access of phones is configured by activating a critical service template when the authentication server becomes unreachable. The voice feature plug-in registers with the Enterprise Policy Manager (EPM) by using an authentication, authorization, and accounting (AAA) voice attribute, and it allows unconditional access to the voice VLAN while the AAA services are unavailable.

To enable critical voice VLAN support, the critical authentication of phones must be configured using a combination of control policy rules and a service template.

When the authentication server is unavailable and the host is unauthorized, the AAA attribute device-traffic-type is not populated. The phone is authorized as an unknown domain, and both the data and voice VLAN are enabled for this device, allowing the device to handle voice traffic.

# How to Configure Critical Voice VLAN Support

## Configuring a Critical Voice VLAN in a Service Template

Perform this task on a port to configure critical voice VLAN support using a service template.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **service-template** *template-name*<br>**Example:**<br>`Device(config)# service-template CRITICAL-DATA` | Defines a template that contains a set of service policy attributes to apply to subscriber sessions and enters service template configuration mode. |
| **Step 4** | **vlan** *vlan-id*<br>**Example:**<br>`Device(config-service-template)# vlan 116` | Assigns a VLAN to a subscriber session. |
| **Step 5** | **exit**<br>**Example:**<br>`Device(config-service-template)# exit` | Exits service template configuration mode and returns to global configuration mode. |
| **Step 6** | **service-template** *template-name*<br>**Example:**<br>`Device(config)# service-template CRITICAL-VOICE` | Defines a template that contains a set of service policy attributes to apply to subscriber sessions and enters service template configuration mode. |
| **Step 7** | **voice vlan**<br>**Example:**<br>`Device(config-service-template)# voice vlan` | Assigns a critical voice VLAN to a subscriber session. |
| **Step 8** | **exit**<br>**Example:**<br>`Device(config-service-template)# exit` | Exits service template configuration mode and returns to global configuration mode. |
| **Step 9** | **class-map type control subscriber** {**match-all** \| **match-any** \| **match-none**} *control-class-name*<br>**Example:** | Creates a control class, which defines the conditions under which the actions of a control policy are executed and enters control class-map filter configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# class-map type control subscriber match-all AAA-SVR-DOWN-UNAUTHD-HOST` | |
| **Step 10** | **match result-type** [**method** {**dot1x** \| **mab** \| **webauth**}] *result-type*<br>**Example:**<br>`Device(config-filter-control-classmap)# match result-type aaa-timeout` | Creates a condition that returns true based on the specified authentication result. |
| **Step 11** | **match authorization-status** {**authorized** \| **unauthorized**}<br>**Example:**<br>`Device(config-filter-control-classmap)# match authorization-status unauthorized` | Creates a condition that returns true based on the authorization status of a session. |
| **Step 12** | **exit**<br>**Example:**<br>`Device(config-filter-control-classmap)# exit` | Exits control class-map filter configuration mode and returns to global configuration mode. |
| **Step 13** | **class-map type control subscriber** {**match-all** \| **match-any** \| **match-none**} *control-class-name*<br>**Example:**<br>`Device(config)# class-map type control subscriber match-all AAA-SVR-DOWN-AUTHD-HOST` | Creates a control class, which defines the conditions under which the actions of a control policy are executed and enters control class-map filter configuration mode. |
| **Step 14** | **match result-type** [**method** {**dot1x** \| **mab** \| **webauth**}] *result-type*<br>**Example:**<br>`Device(config-filter-control-classmap)# match result-type aaa-timeout` | Creates a condition that returns true based on the specified authentication result. |
| **Step 15** | **match authorization-status** {**authorized** \| **unauthorized**}<br>**Example:**<br>`Device(config-filter-control-classmap)# match authorization-status authorized` | Creates a condition that returns true based on the authorization status of a session. |
| **Step 16** | **end**<br>**Example:**<br>`Device(config-filter-control-classmap)# end` | Exits control class-map filter configuration mode and returns to privileged EXEC mode. |

# Activating Critical Voice VLAN

Perform the following task to activate a critical voice VLAN that is configured on a service template.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **policy-map type control subscriber** *control-policy-name*<br>**Example:**<br>`Device(config)# policy-map type control subscriber cisco-subscriber` | Defines a control policy for subscriber sessions and enters control policy-map event configuration mode. |
| **Step 4** | **event authentication-failure** [**match-all** \| **match-first**]<br>**Example:**<br>`Device(config-event-control-policymap)# event authentication-failure match-first` | Specifies the type of event that triggers actions in a control policy if all authentication events are a match and enters control policy-map class configuration mode. |
| **Step 5** | *priority-number* **class** {*control-class-name* \| **always**} [**do-all** \| **do-until-failure** \| **do-until-success**]<br>**Example:**<br>`Device(config-class-control-policymap)# 10 class AAA-SVR-DOWN-UNAUTHD-HOST do-until-failure` | Specifies that the control class should execute the actions in a control policy, in the specified order, until one of the actions fails, and enters control policy-map action configuration mode. |
| **Step 6** | *action-number* **activate** {**policy type control subscriber** *control-policy-name* \| **service-template** *template-name* [**aaa-list** *list-name*] [**precedence** [**replace-all**]]}<br>**Example:**<br>`Device(config-action-control-policymap)# 10 activate service-template CRITICAL-DATA` | Activates a control policy associated with the VLAN on a subscriber session. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | *action-number* **activate** {**policy type control subscriber** *control-policy-name* \| **service-template** *template-name* [**aaa-list** *list-name*] [**precedence** [**replace-all**]]} <br> **Example:** <br> `Device(config-action-control-policymap)# 20 activate service-template CRITICAL-VOICE` | Activates a control policy associated with the voice VLAN on a subscriber session. |
| **Step 8** | *action-number* **authorize** <br> **Example:** <br> `Device(config-action-control-policymap)# 30 authorize` | Initiates the authorization of a subscriber session. |
| **Step 9** | *action-number* **pause reauthentication** <br> **Example:** <br> `Device(config-action-control-policymap)# 40 pause reauthentication` | Pauses the reauthentication process after an authentication failure. |
| **Step 10** | **exit** <br> **Example:** <br> `Device(config-action-control-policymap)# exit` | Exits control policy-map action configuration mode and enters control policy-map class configuration mode. |
| **Step 11** | *priority-number* **class** {*control-class-name* \| **always**} [**do-all** \| **do-until-failure** \| **do-until-success**] <br> **Example:** <br> `Device(config-class-control-policymap)# 20 class AAA-SVR-DOWN-AUTHD-HOST` | Specifies that the control class should execute the actions in a control policy, in the specified order, until one of the actions fails, and enters control policy-map action configuration mode. |
| **Step 12** | *action-number* **pause reauthentication** <br> **Example:** <br> `Device(config-action-control-policymap)# 10 pause reauthentication` | Pauses the reauthentication process after an authentication failure. |
| **Step 13** | **end** <br> **Example:** <br> `Device(config-action-control-policymap)# exit` | Exits control policy-map action configuration mode and enters privileged EXEC mode. |

# Configuration Examples for Critical Voice VLAN Support

## Example: Configuring a Voice VLAN in a Service Template

```
Device> enable
Device# configure terminal
Device(config)# service-template CRITICAL-DATA
Device(config-service-template)# vlan 116
Device(config-service-template)# exit
Device(config)# service-template CRITICAL-VOICE
Device(config-service-template)# voice vlan
Device(config-service-template)# exit
Device(config)# class-map type control subscriber match-all AAA-SVR-DOWN-UNAUTHD-HOST
Device(config-filter-control-classmap)# match result-type aaa-timeout
Device(config-filter-control-classmap)# match authorization-status unauthorized
Device(config-filter-control-classmap)# exit
Device(config)# class-map type control subscriber match-all AAA-SVR-DOWN-AUTHD-HOST
Device(config-filter-control-classmap)# match result-type aaa-timeout
Device(config-filter-control-classmap)# match authorization-status authorized
Device(config-filter-control-classmap)# end
```

## Example: Activating a Critical Voice VLAN on a Service Template

```
Device> enable
Device# configure terminal
Device(config)# policy-map type control subscriber cisco-subscriber
Device(config-event-control-policymap)# event authentication-failure match-first
Device(config-class-control-policymap)# 10 class AAA-SVR-DOWN-UNAUTHD-HOST do-until-failure
Device(config-action-control-policymap)# 10 activate service-template CRITICAL-DATA
Device(config-action-control-policymap)# 10 activate service-template CRITICAL-VOICE
Device(config-action-control-policymap)# 30 authorize
Device(config-action-control-policymap)# 40 pause reauthentication
Device(config-action-control-policymap)# exit
Device(config-class-control-policymap)# 20 class AAA-SVR-DOWN-AUTHD-HOST
Device(config-action-control-policymap)# 10 pause reauthentication
Device(config-action-control-policymap)# end
```

# Feature History for Critical Voice VLAN Support

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

*Table 42: Feature History for Critical Voice VLAN Support*

| Release | Feature Name | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Critical Voice VLAN Support | This feature enables critical voice VLAN support, which puts phone traffic into the configured voice VLAN of a port if the authentication server becomes unreachable.<br><br>Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches |
| Cisco IOS XE Fuji 16.8.1a | Critical Voice VLAN Support | Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 40

# Configuring Local Authentication Using LDAP

This chapter provides information about configuring local authentication for Cisco Identity Based Networking Services.


- Information About Local Authentication Using LDAP, on page 721
- How to Configure Local Authentication Using LDAP, on page 721
- Configuration Examples for Local Authentication Using LDAP, on page 723
- Feature History for Local Authentication Using LDAP, on page 724


# Information About Local Authentication Using LDAP

## Local Authentication Using LDAP

Local authentication using Lightweight Directory Access Protocol (LDAP) allows an endpoint to be authenticated using 802.1X, MAC authentication bypass (MAB), or web authentication with LDAP as a backend. Local authentication in Identity-Based Networking Services also supports associating an authentication, authorization, and accounting (AAA) attribute list with the local username for wireless sessions.

# How to Configure Local Authentication Using LDAP

## Configuring Local Authentication Using LDAP

Perform this task to specify the AAA method list for local authentication and to associate an attribute list with a local username.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables the authentication, authorization, and accounting (AAA) access control model. |
| **Step 4** | **aaa local authentication** {*method-list-name* \| **default**} **authorization** {*method-list-name* \| **default**}<br>**Example:**<br>`Device(config)# aaa local authentication default authorization default` | Specifies the method lists to use for local authentication and authorization from a LDAP server. |
| **Step 5** | **username** *name* **aaa attribute list** *aaa-attribute-list* [**password** *password*]<br>**Example:**<br>`Device(config)# username USER_1 aaa attribute list LOCAL_LIST password CISCO` | Associates a AAA attribute list with a local username. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuring MAC Filtering Support

Perform this task to set the RADIUS compatibility mode, the MAC delimiter, and the MAC address as the username to support MAC filtering.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables the authentication, authorization, and accounting (AAA) access control model. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **aaa group server radius** *group-name*<br>**Example:**<br>`Device(config)# aaa group server radius RAD_GROUP1` | Groups different RADIUS server hosts into distinct lists. |
| **Step 5** | **subscriber mac-filtering security-mode** {**mac** \| **none** \| **shared-secret**}<br>**Example:**<br>`Device(config-sg-radius)# subscriber mac-filtering security-mode mac` | Specifies the RADIUS compatibility mode for MAC filtering.<br>• The default value is **none**. |
| **Step 6** | **mac-delimiter** {**colon** \| **hyphen** \| **none** \| **single-hyphen**}<br>**Example:**<br>`Device(config-sg-radius)# mac-delimiter hyphen` | Specifies the MAC delimiter for RADIUS compatibility mode.<br>• The default value is **none**. |
| **Step 7** | **exit**<br>**Example:**<br>`Device(config-sg-radius)# exit` | Exits server group configuration mode and returns to global configuration mode. |
| **Step 8** | **username** *mac-address* **mac** [**aaa attribute list** *aaa-attribute-list*]<br>**Example:**<br>`Device(config)# username 00-22-WP-EC-23-3C mac aaa attribute list AAA_list1` | Allows a MAC address to be used as the username for MAC filtering done locally. |
| **Step 9** | **exit**<br>**Example:**<br>`Device(config)# exit` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuration Examples for Local Authentication Using LDAP

## Example: Configuring Local Authentication Using LDAP

The following example shows a configuration for local authentication:

```
!
username USER_1 password 0 CISCO
username USER_1 aaa attribute list LOCAL_LIST
aaa new-model
aaa local authentication EAP_LIST authorization EAP_LIST
!
```

# Example: Configuring MAC Filtering Support

The following example shows a configuration for MAC filtering:

```
username 00-22-WP-EC-23-3C mac aaa attribute list AAA_list1
!
aaa new-model
aaa group server radius RAD_GROUP1
subscriber mac-filtering security-mode mac
mac-delimiter hyphen
```

# Feature History for Local Authentication Using LDAP

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

***Table 43: Feature History for Local Authentication Using LDAP***

| Release | Feature Name | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Local Authentication Using LDAP | Introduces support for local authentication using Lightweight Directory Access Protocol (LDAP).<br>Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches |
| Cisco IOS XE Fuji 16.8.1a | Local Authentication Using LDAP | Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 41

# Web Authentication Redirection to Original URL


- Web Authentication Redirection to Original URL Overview , on page 725
- Feature History for Local Authentication Using LDAP, on page 727


## Web Authentication Redirection to Original URL Overview

The Web Authentication Redirection to Original URL feature enables networks to redirect guest users to the URL that they had originally requested. This feature is enabled by default and requires no configuration.

Guest networks are network connections provided by an enterprise to allow their guests to gain access to the Internet and to their own enterprise networks without compromising the security of the host enterprise. Guest users of an enterprise network can connect to the guest access network through either a wired Ethernet connection or a wireless connection.

Guest access uses a captive portal to gather all web requests made by guests and redirect these requests to one of the guest on-boarding web pages. When guests successfully complete the guest workflow, they are redirected to the page that they had originally requested.

The originally requested URL is passed as metadata along with the Cisco Identity Services Engine (ISE) guest access redirect URL. The Cisco ISE is a security policy management and control platform. It automates and simplifies access control and security compliance for wired, wireless, and VPN connectivity. The requested URL is added at the end of the Cisco ISE guest URL so that the device can send the redirect URL to the guest client. The Cisco ISE parses the URL and redirects the guest to the original URL after completing the on-boarding.

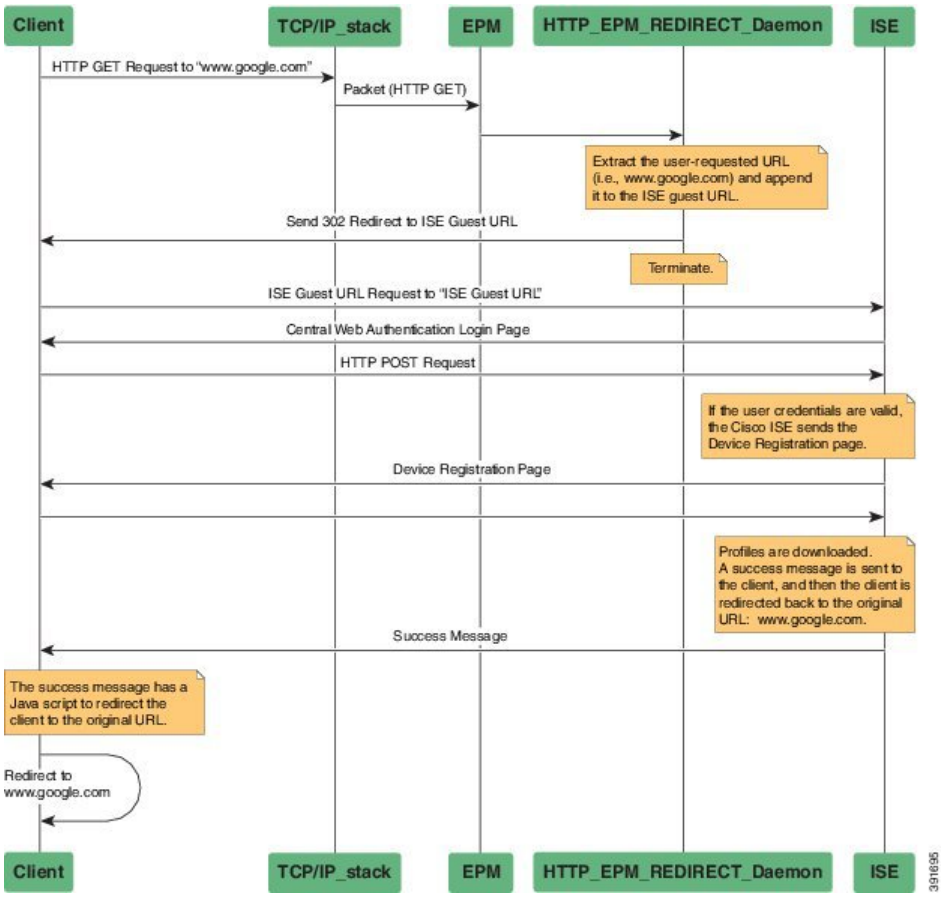
The following is an example of a redirect URL along with the original requested URL:

```
https://10.64.67.92:8443/guestportal/gateway?sessionId=0920269E0000000B0002426B&action=cwa&redirect_
url=http://www.cisco.com/
```

In this example, the URL, https://10.64.67.92:8443/guestportal/gateway?sessionId=0920269E0000000B0002426B&action=cwa is the URL for the guest portal, “&” tells the browser that what follows is a list of name value pairs, and redirect_url=http://www.cisco.com identifies the URL that the user originally requested and to which the user is redirected after completing the guest workflow.

This illustration displays the packet flow that redirects a user to the originally requested URL:

*Figure 46: Original URL Redirection Packet Flow*




1. A user accesses a network for the first time and sends an HTTP request to access www.google.com. When the user first accesses the network, a MAC authentication bypass (MAB) is triggered and the MAC address is sent to the Cisco ISE.

2. The Cisco ISE returns a RADIUS access-accept message (even if the MAC address is not received) along with the redirect access control list (ACL), the ACL-WEBAUTH-REDIRECT message, and the guest web portal URL to the device.

   The RADIUS message instructs the device to open a port that is restricted based on the configured port and the redirect ACLs, for regular network traffic.

3. When the user launches a web browser, the device intercepts the HTTP traffic and redirects the browser to the Cisco ISE central web authentication (CWA) guest web portal URL; the user-requested URL is extracted and appended to the Cisco ISE guest URL.

4. When the user is authenticated, the Cisco ISE sends the Device Registration page to the user. The user enters the required information, and the page is returned to the Cisco ISE. The Cisco ISE downloads user profiles and redirects the user to the originally requested URL <www.google.com>.

# Feature History for Local Authentication Using LDAP

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

***Table 44: Feature History for Local Authentication Using LDAP***

| Release | Feature Name | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Local Authentication Using LDAP | Introduces support for local authentication using Lightweight Directory Access Protocol (LDAP).<br>Support for this feature was introduced only on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches |
| Cisco IOS XE Fuji 16.8.1a | Local Authentication Using LDAP | Support for this feature was introduced only on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 42

# Configuring Port-Based Traffic Control


- Port-Based Traffic Control , on page 729


# Port-Based Traffic Control

Port-based traffic control is a set of Layer 2 features on the Cisco devices used to filter or block packets at the port level in response to specific traffic conditions. The following port-based traffic control features are supported:

- Storm Control
- Protected Ports
- Port Blocking

## Information About Port-Based Traffic Control

### Storm Control

Storm control prevents traffic on a LAN from being disrupted by a broadcast, multicast, or unicast storm on one of the physical interfaces. A LAN storm occurs when packets flood the LAN, creating excessive traffic and degrading network performance. Errors in the protocol-stack implementation, mistakes in network configurations, or users issuing a denial-of-service attack can cause a storm.

Storm control (or traffic suppression) monitors packets passing from an interface to the switching bus and determines if the packet is unicast, multicast, or broadcast. The switch counts the number of packets of a specified type received within the 1-second time interval and compares the measurement with a predefined suppression-level threshold.

With the hardware rate limiter feature of the Storm Control Policer, all broadcast, multicast, and unicast packets with less than 1000 packets per second are blocked. This behavior is due to a hardware limitation on ASIC-based platforms. On these platforms, storm control will work with 1K and above packets per second, and 8K and above bits per second.

#### Measured Traffic Activity

Storm control uses one of these methods to measure traffic activity:

- Bandwidth as a percentage of the total available bandwidth of the port that can be used by the broadcast, multicast, or unicast traffic
- Traffic rate in packets per second at which broadcast, multicast, or unicast packets are received
- Traffic rate in bits per second at which broadcast, multicast, or unicast packets are received

With each method, the port blocks traffic when the rising threshold is reached. The port remains blocked until the traffic rate drops below the falling threshold (if one is specified) and then resumes normal forwarding. If the falling suppression level is not specified, the device blocks all traffic until the traffic rate drops below the rising suppression level. In general, the higher the level, the less effective the protection against broadcast storms.

**Note** When the storm control threshold for any type of traffic is reached, Hardware Rate Limiter drops the traffic exceeding the threshold set. However, the rest of the traffic within the threshold set continues to be forwarded even after the storm is detected. Starting with Cisco IOS XE Gibraltar 16.11.1 release, this behavior is applicable to Cisco Catalyst 9500 Series Switches - High Performance only.

Storm control for unicast is a combination of known unicast and unknown unicast traffic. When storm control for unicast is configured, and it exceeds the configured value, the storm will hit each type of traffic through the hardware policer. The following example describes how the unicast traffic is filtered, when the configured storm is 10%:

- Incoming traffic is unknown unicast 8% + known unicast 7%. Total of 15% storm is not filtered in hardware by the hardware policer.
- Incoming traffic is unknown unicast 11% + known unicast 7%. Total of 18% storm will hit unknown unicast traffic type, and the hardware policer will filter unknown traffic that exceeds 11%.
- Incoming traffic is unknown unicast 11% + known unicast 11%. Total of 22% storm will hit unknown unicast traffic and known unicast traffic, and the hardware policer will filter both unknown and unknown unicast traffic.

**Note** Do not configure both **storm-control unicast** and **storm-control unknown unicast** commands on an interface. If you configure both these commands, it might result in the unknown unicast storm control values to be modified in the hardware.

## Traffic Patterns

Broadcast traffic being forwarded exceeded the configured threshold between time intervals T1 and T2 and between T4 and T5. When the amount of specified traffic exceeds the threshold, all traffic of that kind is dropped for the next time period. Therefore, broadcast traffic is blocked during the intervals following T2 and T5. At the next time interval (for example, T3), if broadcast traffic does not exceed the threshold, it is again forwarded.

The combination of the storm-control suppression level and the 1-second time interval controls the way the storm control algorithm works. A higher threshold allows more packets to pass through. A threshold value of 100 percent means that no limit is placed on the traffic. A value of 0.0 means that all broadcast, multicast, or unicast traffic on that port is blocked.

**Note** Because packets do not arrive at uniform intervals, the 1-second time interval during which traffic activity is measured can affect the behavior of storm control.

You use the **storm-control** interface configuration commands to set the threshold value for each traffic type.

### Storm Control Using a Hardware Rate Limiter

Traffic storm control monitors incoming traffic levels over a configured interval. However, the reaction time taken by storm control is slightly slower as it is based on statistics counters to identify a storm. With the hardware rate limiter, the action is taken at the ASIC level, and as a result, the storm control action starts immediately; as soon as the traffic rate reaches the set threshold level. The hardware rate limiter implements policers for broadcast, multicast, unicast, and unknown unicast traffic.

## Protected Ports

Some applications require that no traffic be forwarded at Layer 2 between ports on the same device so that one neighbor does not see the traffic generated by another neighbor. In such an environment, the use of protected ports ensures that there is no exchange of unicast, broadcast, or multicast traffic between these ports on the device.

Protected ports have these features:

- A protected port does not forward any traffic (unicast, multicast, or broadcast) to any other port that is also a protected port. Data traffic cannot be forwarded between protected ports at Layer 2; only control traffic, such as PIM packets, is forwarded because these packets are processed by the CPU and forwarded in software. All data traffic passing between protected ports must be forwarded through a Layer 3 device.
- Forwarding behavior between a protected port and a nonprotected port proceeds as usual.

### Protected Ports Guidelines

You can configure protected ports on a physical interface (for example, Gigabit Ethernet port 1) or an EtherChannel group (for example, port-channel 5). When you enable protected ports for a port channel, it is enabled for all ports in the port-channel group.

By default no protected ports are defined.

## Port Blocking

By default, the device floods packets with unknown destination MAC addresses out of all ports. If unknown unicast and multicast traffic is forwarded to a protected port, there could be security issues. To prevent unknown unicast or multicast traffic from being forwarded from one port to another, you can block a port (protected or nonprotected) from flooding unknown unicast or multicast packets to other ports.

**Note** With multicast traffic, the port blocking feature blocks only pure Layer 2 packets. Multicast packets that contain IPv4 or IPv6 information in the header are not blocked.

# How to Configure Port-Based Traffic Control

## Configuring Storm Control and Threshold Levels

You configure storm control on a port and enter the threshold level that you want to be used for a particular type of traffic.

However, because of hardware limitations and the way in which packets of different sizes are counted, threshold percentages are approximations. Depending on the sizes of the packets making up the incoming traffic, the actual enforced threshold might differ from the configured level by several percentage points.

**Note** Storm control is supported on physical interfaces. You can also configure storm control on an EtherChannel. When storm control is configured on an EtherChannel, the storm control settings propagate to the EtherChannel physical interfaces.

Follow these steps to storm control and threshold levels:

### Before you begin

Storm control is supported on physical interfaces. You can also configure storm control on an EtherChannel. When storm control is configured on an EtherChannel, the storm control settings propagate to the EtherChannel physical interfaces.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet1/0/1` | Specifies the interface to be configured, and enters interface configuration mode. |
| **Step 4** | **storm-control** {**broadcast** \| **multicast** \| **unicast**} **level** {*level* [*level-low*] \| **bps** *bps* [*bps-low*] \| **pps** *pps* [*pps-low*]}<br>**Example:** | Configures broadcast, multicast, or unicast storm control. By default, storm control is disabled.<br>• For *level*, specifies the rising threshold level for broadcast, multicast, or unicast |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-if)# storm-control unicast level 87 65` | traffic as a percentage (up to two decimal places) of the bandwidth. The port blocks traffic when the rising threshold is reached. The range is 0.00 to 100.00.<br>• (Optional) For *level-low*, specifies the falling threshold level as a percentage (up to two decimal places) of the bandwidth. This value must be less than or equal to the rising suppression value. The port forwards traffic when traffic drops below this level. If you do not configure a falling suppression level, it is set to the rising suppression level. The range is 0.00 to 100.00.<br>If you set the threshold to the maximum value (100 percent), no limit is placed on the traffic. If you set the threshold to 0.0, all broadcast, multicast, and unicast traffic on that port is blocked.<br>• For **bps** *bps*, specifies the rising threshold level for broadcast, multicast, or unicast traffic in bits per second (up to one decimal place). The port blocks traffic when the rising threshold is reached. The range is 0.0 to 10000000000.0.<br>• (Optional) For *bps-low*, specifies the falling threshold level in bits per second (up to one decimal place). It can be less than or equal to the rising threshold level. The port forwards traffic when traffic drops below this level. The range is 0.0 to 10000000000.0.<br>• For **pps** *pps*, specifies the rising threshold level for broadcast, multicast, or unicast traffic in packets per second (up to one decimal place). The port blocks traffic when the rising threshold is reached. The range is 0.0 to 10000000000.0.<br>• (Optional) For *pps-low*, specifies the falling threshold level in packets per second (up to one decimal place). It can be less than or equal to the rising threshold level. The port forwards traffic when traffic drops below this level. The range is **0.0 to** 10000000000.0. |

| | Command or Action | Purpose |
|---|---|---|
| | | For BPS and PPS settings, you can use metric suffixes such as k, m, and g for large number thresholds. |
| **Step 5** | **storm-control action** {**shutdown** \| **trap**}<br>**Example:**<br>`Device(config-if)# storm-control action trap` | Specifies the action to be taken when a storm is detected. Once a storm is detected, the **shutdown** or **trap** action is applied on all the traffic. The default is to filter out the traffic and not to send traps.<br>• Select the **shutdown** keyword to error-disable the port during a storm.<br>• Select the **trap** keyword to generate an SNMP trap when a storm is detected. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 7** | **show storm-control** [*interface-id*] [**broadcast** \| **multicast** \| **unicast**]<br>**Example:**<br>`Device# show storm-control gigabitethernet1/0/1 unicast` | Verifies the storm control suppression levels set on the interface for the specified traffic type. If you do not enter a traffic type, details for all traffic types (broadcast, multicast and unicast) are displayed. |

## Configuring a Protected Port

### Before you begin

Protected ports are not pre-defined. This is the task to configure one.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Specifies the interface to be configured, and enters interface configuration mode. |
| **Step 4** | **switchport protected**<br>**Example:**<br>`Device(config-if)# switchport protected` | Configures the interface to be a protected port. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

### Monitoring Protected Ports

*Table 45: Commands for Displaying Protected Port Settings*

| Command | Purpose |
|---|---|
| **show interfaces** [*interface-id*] **switchport** | Displays the administrative and operational status of all sw (nonrouting) ports or the specified port, including port bloc protection settings. |

## Blocking Flooded Traffic on an Interface

### Before you begin

The interface can be a physical interface or an EtherChannel group. When you block multicast or unicast traffic for a port channel, it is blocked on all ports in the port-channel group.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device# configure terminal` | |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Specifies the interface to be configured, and enter interface configuration mode. |
| **Step 4** | **switchport block multicast**<br>**Example:**<br>`Device(config-if)# switchport block multicast` | Blocks unknown multicast forwarding out of the port. |
| **Step 5** | **switchport block unicast**<br>**Example:**<br>`Device(config-if)# switchport block unicast` | Blocks unknown unicast forwarding out of the port. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |

### Monitoring Port Blocking

*Table 46: Commands for Displaying Port Blocking Settings*

| Command | Purpose |
|---|---|
| **show interfaces** [*interface-id*] **switchport** | Displays the administrative and operational status of all switch (nonrouting) ports or the specified port, including port blockin protection settings. |

# Additional References for Port-Based Traffic Control

### Related Documents

| Related Topic | Document Title |
|---|---|
| Port Security | Port Security chapter in the *Security Configuration Guide* |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br><br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br><br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/support |

# Feature History for Port-Based Traffic Control

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Port-Based Traffic Control | Port-based traffic control is a set of Layer 2 features on the Cisco Catalyst switches used to filter or block packets at the port level in response to specific traffic conditions.<br><br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Port-Based Traffic Control | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Gibraltar 16.11.1 | Storm Control with a Hardware Rate Limiter | With the hardware rate limiter, the storm control action is taken at the ASIC level, and as a result, the action starts immediately; as soon as the traffic rate reaches the set threshold level.<br><br>Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 43

# Port Security


- Prerequisites for Port Security, on page 739
- Restrictions for Port Security, on page 739
- Information About Port Security, on page 740
- How to Configure Port Security, on page 744
- Configuration Examples for Port Security, on page 752
- Feature History for Port Security, on page 752


## Prerequisites for Port Security

If you try to set the maximum value to a number less than the number of secure addresses already configured on an interface, the command is rejected.

## Restrictions for Port Security

- The maximum number of secure MAC addresses that you can configure on a switch is set by the maximum number of available MAC addresses allowed in the system. This number is the total of available MAC addresses, including those used for other Layer 2 functions and any other secure MAC addresses configured on interfaces.
- Port Security is not supported on EtherChanel interfaces.
- Port Security is not supported on private VLAN ports.
- We recommend that you do not enable port security on an 802.1X authenticator interface.

  When port-security is disabled on a port, the 802.1X sessions on the port get removed, because the aging timer and inactivity type is still configured. To ensure that the 802.1X sessions are not removed, when disabling port-security, disable the aging timer and inactivity type by removing the following commands:

  - **switchport port-security aging time 1**
  - **switchport port-security aging type inactivity**

  If the inactivity timer is required, see the section "Enabling and Configuring Port Security Aging".

# Information About Port Security

## Port Security

You can use the port security feature to restrict input to an interface by limiting and identifying MAC addresses of the stations allowed to access the port. When you assign secure MAC addresses to a secure port, the port does not forward packets with source addresses outside the group of defined addresses. If you limit the number of secure MAC addresses to one and assign a single secure MAC address, the workstation attached to that port is assured the full bandwidth of the port.

If a port is configured as a secure port and the maximum number of secure MAC addresses is reached, when the MAC address of a station attempting to access the port is different from any of the identified secure MAC addresses, a security violation occurs. Also, if a station with a secure MAC address configured or learned on one secure port attempts to access another secure port, a violation is flagged.

## Types of Secure MAC Addresses

The switch supports these types of secure MAC addresses:

- Static secure MAC addresses—These are manually configured by using the **switchport port-security mac-address** *mac-address* interface configuration command, stored in the address table, and added to the switch running configuration.
- Dynamic secure MAC addresses—These are dynamically configured, stored only in the address table, and removed when the switch restarts.
- Sticky secure MAC addresses—These can be dynamically learned or manually configured, stored in the address table, and added to the running configuration. If these addresses are saved in the configuration file, when the switch restarts, the interface does not need to dynamically reconfigure them.

## Default MAC Address Table Settings

The following table shows the default settings for the MAC address table.

***Table 47: Default Settings for the MAC Address***

| Feature | Default Setting |
|---|---|
| Aging time | 300 seconds |
| Dynamic addresses | Automatically learned |
| Static addresses | None configured |

## MAC Address Table Creation

With multiple MAC addresses supported on all ports, you can connect any port on the device to other network devices. The device provides dynamic addressing by learning the source address of packets it receives on

each port and adding the address and its associated port number to the address table. As devices are added or removed from the network, the device updates the address table, adding new dynamic addresses and aging out those that are not in use.

The aging interval is globally configured. However, the device maintains an address table for each VLAN, and STP can accelerate the aging interval on a per-VLAN basis.

The device sends packets between any combination of ports, based on the destination address of the received packet. Using the MAC address table, the device forwards the packet only to the port associated with the destination address. If the destination address is on the port that sent the packet, the packet is filtered and not forwarded. The device always uses the store-and-forward method: complete packets are stored and checked for errors before transmission.

# Sticky Secure MAC Addresses

You can configure an interface to convert the dynamic MAC addresses to sticky secure MAC addresses and to add them to the running configuration by enabling sticky learning. The interface converts all the dynamic secure MAC addresses, including those that were dynamically learned before sticky learning was enabled, to sticky secure MAC addresses. All sticky secure MAC addresses are added to the running configuration.

The sticky secure MAC addresses do not automatically become part of the configuration file, which is the startup configuration used each time the switch restarts. If you save the sticky secure MAC addresses in the configuration file, when the switch restarts, the interface does not need to relearn these addresses. If you do not save the sticky secure addresses, they are lost.

If sticky learning is disabled, the sticky secure MAC addresses are converted to dynamic secure addresses and are removed from the running configuration.

# Security Violations

It is a security violation when one of these situations occurs:

- The maximum number of secure MAC addresses have been added to the address table, and a station whose MAC address is not in the address table attempts to access the interface.
- An address learned or configured on one secure interface is seen on another secure interface in the same VLAN.
- Running diagnostic tests with port security enabled.

You can configure the interface for one of three violation modes, based on the action to be taken if a violation occurs:

- protect—when the number of secure MAC addresses reaches the maximum limit allowed on the port, packets with unknown source addresses are dropped until you remove a sufficient number of secure MAC addresses to drop below the maximum value or increase the number of maximum allowable addresses. You are not notified that a security violation has occurred.

**Note** We do not recommend configuring the protect violation mode on a trunk port. The protect mode disables learning when any VLAN reaches its maximum limit, even if the port has not reached its maximum limit.

- restrict—when the number of secure MAC addresses reaches the maximum limit allowed on the port, packets with unknown source addresses are dropped until you remove a sufficient number of secure MAC addresses to drop below the maximum value or increase the number of maximum allowable addresses. In this mode, you are notified that a security violation has occurred. An SNMP trap is sent, a syslog message is logged, and the violation counter increments.
- shutdown—a port security violation causes the interface to become error-disabled and to shut down immediately, and the port LED turns off. When a secure port is in the error-disabled state, you can bring it out of this state by entering the **errdisable recovery cause** *psecure-violation* global configuration command, or you can manually re-enable it by entering the **shutdown** and **no shut down** interface configuration commands. This is the default mode.
- shutdown vlan—Use to set the security violation mode per-VLAN. In this mode, the VLAN is error disabled instead of the entire port when a violation occurs

This table shows the violation mode and the actions taken when you configure an interface for port security.

***Table 48: Security Violation Mode Actions***

| Violation Mode | Traffic is forwarded [6] | Sends SNMP trap | Sends syslog message | Displays error message [7] | Violation counter increments | Shuts d |
|---|---|---|---|---|---|---|
| protect | No | No | No | No | No | No |
| restrict | No | Yes | Yes | No | Yes | No |
| shutdown | No | No | No | No | Yes | Yes |
| shutdown vlan | No | No | Yes | No | Yes | No [8] |

[6] Packets with unknown source addresses are dropped until you remove a sufficient number of secure MAC addresses.
[7] The switch returns an error message if you manually configure an address that would cause a security violation.
[8] Shuts down only the VLAN on which the violation occurred.

# Port Security Aging

You can use port security aging to set the aging time for all secure addresses on a port. Two types of aging are supported per port:

- Absolute—The secure addresses on the port are deleted after the specified aging time.
- Inactivity—The secure addresses on the port are deleted only if the secure addresses are inactive for the specified aging time.

# Default Port Security Configuration

*Table 49: Default Port Security Configuration*

| Feature | Default Setting |
|---|---|
| Port security | Disabled on a port. |
| Sticky address learning | Disabled. |
| Maximum number of secure MAC addresses per port | One address |
| Violation mode | Shutdown. The port shuts down when the maximum number of secure MAC addresses is exceeded. |
| Port security aging | Disabled. Aging time is 0.<br>Static aging is disabled.<br>Type is absolute. |

# Port Security Configuration Guidelines

- Port security can only be configured on static access ports or trunk ports. A secure port cannot be a dynamic access port.
- A secure port cannot be a destination port for Switched Port Analyzer (SPAN).
- Voice VLAN is only supported on access ports and not on trunk ports, even though the configuration is allowed.
- When you enable port security on an interface that is also configured with a voice VLAN, set the maximum allowed secure addresses on the port to two. When the port is connected to a Cisco IP phone, the IP phone requires one MAC address. The Cisco IP phone address is learned on the voice VLAN, but is not learned on the access VLAN. If you connect a single PC to the Cisco IP phone, no additional MAC addresses are required. If you connect more than one PC to the Cisco IP phone, you must configure enough secure addresses to allow one for each PC and one for the phone.
- When a trunk port configured with port security and assigned to an access VLAN for data traffic and to a voice VLAN for voice traffic, entering the **switchport voice** and **switchport priority extend** interface configuration commands has no effect.

  When a connected device uses the same MAC address to request an IP address for the access VLAN and then an IP address for the voice VLAN, only the access VLAN is assigned an IP address.
- When you enter a maximum secure address value for an interface, and the new value is greater than the previous value, the new value overwrites the previously configured value. If the new value is less than the previous value and the number of configured secure addresses on the interface exceeds the new value, the command is rejected.
- The switch does not support port security aging of sticky secure MAC addresses.

This table summarizes port security compatibility with other port-based features.

*Table 50: Port Security Compatibility with Other Switch Features*

| Type of Port or Feature on Port | Compatible with Port Security |
|---|---|
| DTP [9] port [10] | No |
| Trunk port | Yes |
| Dynamic-access port [11] | No |
| Routed port | No |
| SPAN source port | Yes |
| SPAN destination port | No |
| EtherChannel | No |
| Tunneling port | Yes |
| Protected port | Yes |
| IEEE 802.1x port | Yes |
| Voice VLAN port [12] | Yes |
| IP source guard | Yes |
| Dynamic Address Resolution Protocol (ARP) inspection | Yes |
| Flex Links | Yes |

9 DTP=Dynamic Trunking Protocol
10 A port configured with the **switchport mode dynamic** interface configuration command.
11 A VLAN Query Protocol (VQP) port configured with the **switchport access vlan dynamic** interface configuration command.
12 You must set the maximum allowed secure addresses on the port to two plus the maximum number of secure addresses allowed on the access VLAN.

# How to Configure Port Security

## Enabling and Configuring Port Security

### Before you begin

This task restricts input to an interface by limiting and identifying MAC addresses of the stations allowed to access the port:

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Specifies the interface to be configured, and enter interface configuration mode. |
| **Step 4** | **switchport mode** {**access** \| **trunk**}<br>**Example:**<br>`Device(config-if)# switchport mode access` | Sets the interface switchport mode as access or trunk; an interface in the default mode (dynamic auto) cannot be configured as a secure port. |
| **Step 5** | **switchport voice vlan** *vlan-id*<br>**Example:**<br>`Device(config-if)# switchport voice vlan 22` | Enables voice VLAN on a port.<br>*vlan-id*—Specifies the VLAN to be used for voice traffic. |
| **Step 6** | **switchport port-security**<br>**Example:**<br>`Device(config-if)# switchport port-security` | Enables port security on the interface.<br>**Note**<br>Under certain conditions, when port security is enabled on the member ports in a switch stack, the DHCP and ARP packets would be dropped. As a workaround, shutdown the interface and then configure the **no shutdown** command. |
| **Step 7** | **switchport port-security [maximum** *value* [**vlan** {*vlan-list* \| {**access** \| **voice**}}]]<br>**Example:**<br>`Device(config-if)# switchport` | (Optional) Sets the maximum number of secure MAC addresses for the interface. The maximum number of secure MAC addresses that you can configure on a switch or switch stack is set by the maximum number of available MAC addresses allowed in the |

| | Command or Action | Purpose |
|---|---|---|
| | `port-security maximum 20` | system. This number is the total of available MAC addresses, including those used for other Layer 2 functions and any other secure MAC addresses configured on interfaces.<br><br>(Optional) **vlan**—sets a per-VLAN maximum value<br><br>Enter one of these options after you enter the **vlan** keyword:<br><br>• *vlan-list*—On a trunk port, you can set a per-VLAN maximum value on a range of VLANs separated by a hyphen or a series of VLANs separated by commas. For nonspecified VLANs, the per-VLAN maximum value is used.<br><br>• **access**—On an access port, specifies the VLAN as an access VLAN.<br><br>• **voice**—On an access port, specifies the VLAN as a voice VLAN.<br><br>**Note**<br>The **voice** keyword is available only if a voice VLAN is configured on a port and if that port is not the access VLAN. If an interface is configured for voice VLAN, configure a maximum of two secure MAC addresses. |
| **Step 8** | **switchport port-security violation** {**protect** \| **restrict** \| **shutdown** \| **shutdown vlan**}<br><br>**Example:**<br><br>`Device(config-if)# switchport port-security violation restrict` | (Optional) Sets the violation mode, the action to be taken when a security violation is detected, as one of these:<br><br>• **protect**—When the number of port secure MAC addresses reaches the maximum limit allowed on the port, packets with unknown source addresses are dropped until you remove a sufficient number of secure MAC addresses to drop below the maximum value or increase the number of maximum allowable addresses. You are not notified that a security violation has occurred.<br><br>**Note**<br>We do not recommend configuring the protect mode on a trunk port. The protect mode disables learning when any VLAN reaches its maximum limit, even if the port has not reached its maximum limit. |

| | Command or Action | Purpose |
|---|---|---|
| | | • **restrict**—When the number of secure MAC addresses reaches the limit allowed on the port, packets with unknown source addresses are dropped until you remove a sufficient number of secure MAC addresses or increase the number of maximum allowable addresses. An SNMP trap is sent, a syslog message is logged, and the violation counter increments.<br>• **shutdown**—The interface is error-disabled when a violation occurs, and the port LED turns off. An SNMP trap is sent, a syslog message is logged, and the violation counter increments.<br>• **shutdown vlan**—Use to set the security violation mode per VLAN. In this mode, the VLAN is error disabled instead of the entire port when a violation occurs.<br>**Note**<br>When a secure port is in the error-disabled state, you can bring it out of this state by entering the **errdisable recovery cause psecure-violation** global configuration command. You can manually re-enable it by entering the **shutdown** and **no shutdown** interface configuration commands or by using the **clear errdisable interface vlan** privileged EXEC command. |
| **Step 9** | **switchport port-security [mac-address** *mac-address* [**vlan** {*vlan-id* \| {**access** \| **voice**}}]<br>**Example:**<br>`DEvice(config-if)# switchport port-security mac-address 00:A0:C7:12:C9:25 vlan 3 voice` | (Optional) Enters a secure MAC address for the interface. You can use this command to enter the maximum number of secure MAC addresses. If you configure fewer secure MAC addresses than the maximum, the remaining MAC addresses are dynamically learned.<br>**Note**<br>If you enable sticky learning after you enter this command, the secure addresses that were dynamically learned are converted to sticky secure MAC addresses and are added to the running configuration.<br>(Optional) **vlan**—sets a per-VLAN maximum value. |

| | Command or Action | Purpose |
|---|---|---|
| | | Enter one of these options after you enter the **vlan** keyword:<br>• *vlan-id*—On a trunk port, you can specify the VLAN ID and the MAC address. If you do not specify a VLAN ID, the native VLAN is used.<br>• **access**—On an access port, specifies the VLAN as an access VLAN.<br>• **voice**—On an access port, specifies the VLAN as a voice VLAN.<br>**Note**<br>The **voice** keyword is available only if a voice VLAN is configured on a port and if that port is not the access VLAN. If an interface is configured for voice VLAN, configure a maximum of two secure MAC addresses. |
| **Step 10** | **switchport port-security mac-address sticky**<br>**Example:**<br>`Device(config-if)# switchport port-security mac-address sticky` | (Optional) Enables sticky learning on the interface. |
| **Step 11** | **switchport port-security mac-address sticky** [*mac-address* \| **vlan** {*vlan-id* \| {**access** \| **voice**}}]<br>**Example:**<br>`Device(config-if)# switchport port-security mac-address sticky 00:A0:C7:12:C9:25 vlan voice` | (Optional) Enters a sticky secure MAC address, repeating the command as many times as necessary. If you configure fewer secure MAC addresses than the maximum, the remaining MAC addresses are dynamically learned, are converted to sticky secure MAC addresses, and are added to the running configuration.<br>**Note**<br>If you do not enable sticky learning before this command is entered, an error message appears, and you cannot enter a sticky secure MAC address.<br>(Optional) **vlan**—sets a per-VLAN maximum value.<br>Enter one of these options after you enter the **vlan** keyword:<br>• *vlan-id*—On a trunk port, you can specify the VLAN ID and the MAC address. If you do not specify a VLAN ID, the native VLAN is used. |

| | Command or Action | Purpose |
|---|---|---|
| | | • **access**—On an access port, specifies the VLAN as an access VLAN.<br>• **voice**—On an access port, specifies the VLAN as a voice VLAN.<br>**Note**<br>The **voice** keyword is available only if a voice VLAN is configured on a port and if that port is not the access VLAN. |
| **Step 12** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 13** | **show port-security**<br>**Example:**<br>`Device# show port-security` | Displays information about the port-security settings. |

# Enabling and Configuring Port Security Aging

Use this feature to remove and add devices on a secure port without manually deleting the existing secure MAC addresses and to still limit the number of secure addresses on a port. You can enable or disable the aging of secure addresses on a per-port basis.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-id*<br>**Example:**<br>`Device(config)# interface` | Specifies the interface to be configured, and enter interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `gigabitethernet1/0/1` | |
| **Step 4** | **switchport port-security aging** {**static** \| **time** *time* \| **type** {**absolute** \| **inactivity**}}<br>**Example:**<br>`Device(config-if)# switchport port-security aging time 120` | Enables or disable static aging for the secure port, or set the aging time or type.<br>**Note**<br>The switch does not support port security aging of sticky secure addresses.<br>Enter **static** to enable aging for statically configured secure addresses on this port.<br>For *time*, specifies the aging time for this port. The valid range is from 0 to 1440 minutes.<br>For **type**, select one of these keywords:<br>• **absolute**—Sets the aging type as absolute aging. All the secure addresses on this port age out exactly after the time (minutes) specified lapses and are removed from the secure address list.<br>• **inactivity**—Sets the aging type as inactivity aging. The secure addresses on this port age out only if there is no data traffic from the secure source addresses for the specified time period. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config-if)# end` | Exits interface configuration mode and returns to privileged EXEC mode. |
| **Step 6** | **show port-security** [**interface** *interface-id*] [**address**]<br>**Example:**<br>`Device# show port-security interface gigabitethernet1/0/1` | Displays information about the port-security settings on the specified interface. |

# Changing the Address Aging Time

Follow these steps to configure the dynamic address table aging time:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** | Enables privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device> enable` | • Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **mac address-table aging-time** [*0* \| *10-1000000*] [**routed-mac** \| **vlan** *vlan-id*]<br>**Example:**<br>`Device(config)# mac address-table aging-time 500 vlan 2` | Sets the length of time that a dynamic entry remains in the MAC address table after the entry is used or updated.<br>The range is 10 to 1000000 seconds. The default is 300. You can also enter 0, which disables aging. Static address entries are never aged or removed from the table.<br>*vlan-id*—Valid IDs are 1 to 4094. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Monitoring Port Security

This table displays port security information.

***Table 51: Commands for Displaying Port Security Status and Configuration***

| Command | Purpose |
|---|---|
| **show port-security** [**interface** *interface-id*] | Displays port security settings for the device or for the speci including the maximum allowed number of secure MAC ad each interface, the number of secure MAC addresses on the number of security violations that have occurred, and the vi |
| **show port-security** [**interface** *interface-id*] **address** | Displays all secure MAC addresses configured on all device on a specified interface with aging information for each add |
| **show port-security interface** *interface-id* **vlan** | Displays the number of secure MAC addresses configured the specified interface. |

# Configuration Examples for Port Security

This example shows how to enable port security on a port and to set the maximum number of secure addresses to 50. The violation mode is the default, no static secure MAC addresses are configured, and sticky learning is enabled.

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet1/0/1
Device(config-if)# switchport mode access
Device(config-if)# switchport port-security
Device(config-if)# switchport port-security maximum 50
Device(config-if)# switchport port-security mac-address sticky
Device(config-if)# end
```

This example shows how to configure a static secure MAC address on VLAN 3 on a port:

```
Device> enable
Device# configure terminal
Device(config)# interface gigabitethernet1/0/2
Device(config-if)# switchport mode trunk
Device(config-if)# switchport port-security
Device(config-if)# switchport port-security mac-address 0000.0200.0004 vlan 3
Device(config-if)# end
```

This example shows how to enable sticky port security on a port, to manually configure MAC addresses for data VLAN and voice VLAN, and to set the total maximum number of secure addresses to 20 (10 for data VLAN and 10 for voice VLAN).

```
Device> enable
Device# configure terminal
Device(config)# interface tengigabitethernet1/0/1
Device(config-if)# switchport access vlan 21
Device(config-if)# switchport mode access
Device(config-if)# switchport voice vlan 22
Device(config-if)# switchport port-security
Device(config-if)# switchport port-security maximum 20
Device(config-if)# switchport port-security violation restrict
Device(config-if)# switchport port-security mac-address sticky
Device(config-if)# switchport port-security mac-address sticky 0000.0000.0002
Device(config-if)# switchport port-security mac-address 0000.0000.0003
Device(config-if)# switchport port-security mac-address sticky 0000.0000.0001 vlan voice
Device(config-if)# switchport port-security mac-address 0000.0000.0004 vlan voice
Device(config-if)# switchport port-security maximum 10 vlan access
Device(config-if)# switchport port-security maximum 10 vlan voice
Device(config-if)# end
```

# Feature History for Port Security

This table provides release and related information for features explained in this module.

These features are available on all releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Port Security | The Port Security feature restricts input to an interface by limiting and identifying MAC addresses of the stations allowed to access the port.<br><br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Port Security | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Everest 16.5.1a | Port Security MAC Aging | When devices are added or removed from a network, the device updates the address table, adding new dynamic addresses and aging out those that are not in use.<br><br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Port Security MAC Aging | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 44

# Configuring Control Plane Policing


- Restrictions for CoPP, on page 755
- Information About CoPP, on page 756
- How to Configure CoPP, on page 765
- Configuration Examples for CoPP, on page 768
- Monitoring CoPP, on page 773
- Feature History for CoPP, on page 773


## Restrictions for CoPP

Restrictions for control plane policing (CoPP) include the following:

- Only ingress CoPP is supported. The **system-cpp-policy** policy-map is available on the control plane interface, and only in the ingress direction.
- Only the **system-cpp-policy** policy-map can be installed on the control plane interface.
- The **system-cpp-policy** policy-map and the system-defined classes cannot be modified or deleted.
- Only the **police** action is allowed under the **system-cpp-policy** policy-map. The police rate for system-defined classes must be configured only in packets per second (pps).
- One or more CPU queues are part of each class-map. Where multiple CPU queues belong to one class-map, changing the policer rate of a class-map affects all CPU queues that belong to that class-map. Similarly, disabling the policer in a class-map disables all queues that belong to that class-map. See *Table: System-Defined Values for CoPP* for information about which CPU queues belong to each class-map.
- We recommend not disabling the policer for a system-defined class map, that is, do not configure **no police rate** *rate* **pps** command. Doing so affects the overall system health in case of high traffic towards the CPU. Further, even if you disable the policer rate for a system-defined class map, the systems automatically reverts to the default policer rate after system bootup in order to protect the system bring-up process.
- The **show run** command does not display information about classes configured under `system-cpp policy`, when they are left at default values. Use the **show policy-map system-cpp-policy** or the **show policy-map control-plane** commands instead.

  You can continue use the **show run** command to display information about custom policies.

- A protocol with a huge number of CPU-bound packets may impact other protocols in the same class, as some of these protocols share the same policer. For example, Address Resolution Protocol (ARP) shares 4000 hardware policers with an array of host protocols like Telnet, Internet Control Message Protocol (ICMP), SSH, FTP, and SNMP in the system-cpp-police-forus class. If there is an ARP poisoning or an ICMP attack, hardware policers start throttling any incoming traffic that exceeds 4000 packets per second to protect the CPU and the overall integrity of the system. As a result, ARP and ICMP host protocols are dropped, along with any other host protocols that share the same class.
- Starting from Cisco IOS XE Fuji 16.8.1a, the creation of user-defined class-maps is not supported.

# Information About CoPP

This chapter describes how control plane policing (CoPP) works on your device and how to configure it.

## CoPP Overview

The CoPP feature improves security on your device by protecting the CPU from unnecessary traffic and denial of service (DoS) attacks. It can also protect control traffic and management traffic from traffic drops caused by high volumes of other, lower priority traffic.

Your device is typically segmented into three planes of operation, each with its own objective:

- The data plane, to forward data packets.
- The control plane, to route data correctly.
- The management plane, to manage network elements.

You can use CoPP to protect most of the CPU-bound traffic and ensure routing stability, reachability, and packet delivery. Most importantly, you can use CoPP to protect the CPU from a DoS attack.

CoPP uses the modular QoS command-line interface (MQC) and CPU queues to achieve these objectives. Different types of control plane traffic are grouped together based on certain criteria, and assigned to a CPU queue. You can manage these CPU queues by configuring dedicated policers in hardware. For example, you can modify the policer rate for certain CPU queues (traffic-type), or you can disable the policer for a certain type of traffic.

Although the policers are configured in hardware, CoPP does not affect CPU performance or the performance of the data plane. But since it limits the number of packets going to CPU, the CPU load is controlled. This means that services waiting for packets from hardware may see a more controlled rate of incoming packets (the rate being user-configurable).

## System-Defined Aspects of CoPP

When you power-up the device for the first time, the system automatically performs the following tasks:

- Looks for policy-map **system-cpp-policy**. If not found, the system creates and installs it on the control-plane.
- Creates eighteen class-maps under **system-cpp-policy**.

  The next time you power-up the device, the system detects the policy and class maps that have already been created.

- Enables all CPU queues by default, with their respective default rate. The default rates are indicated in the table System-Defined Values for CoPP.

The **system-cpp-policy** policy map is a system-default policy map, and normally, you do not have to expressly save it to the startup configuration of the device. But, a *failed* bulk synchronization with a standby device can result in the configuration being erased from the startup configuration. In case this happens, you have to manually save the **system-cpp-policy** policy map to the startup configuration. Use the **show running-config** privileged EXEC command to verify that it has been saved:

```
policy-map system-cpp-policy
```

The following table (System-Defined Values for CoPP) lists the class-maps that the system creates when you load the device. It lists the policer that corresponds to each class-map and one or more CPU queues that are grouped under each class-map. There is a one-to-one mapping of class-maps to policers; and one or more CPU queues map to a class-map. This is followed by another table (CPU Queues and Associated Features), which lists features associated with each CPU queue.

***Table 52: System-Defined Values for CoPP***

| Class Maps Names | Policer Index (Policer No.) | CPU queues (Queue No.) |
|---|---|---|
| system-cpp- police-data | WK_CPP_POLICE_DATA(0) | WK_CPU_Q_ICMP_GEN(3)<br>WK_CPU_Q_BROADCAST(12)<br>WK_CPU_Q_ICMP_REDIRECT(6) |
| system-cpp-police-l2- control | WK_CPP_POLICE_L2_ CONTROL(1) | WK_CPU_Q_L2_CONTROL(1) |
| system-cpp-police-routing-control | WK_CPP_POLICE_ROUTING_CONTROL(2) | WK_CPU_Q_ROUTING_CONTROL(4)<br>WK_CPU_Q_LOW_LATENCY (27) |
| system-cpp-police-punt-webauth | WK_CPP_POLICE_PU NT_WEBAUTH(7) | WK_CPU_Q_PUNT_WEBAUTH(22) |
| system-cpp-police- topology-control | WK_CPP_POLICE_TOPOLOGY_CONTROL(8) | WK_CPU_Q_TOPOLOGY_CONTROL(15) |
| system-cpp-police- multicast | WK_CPP_POLICE_MULTICAST(9) | WK_CPU_Q_TRANSIT_TRAFFIC(18)<br>WK_CPU_Q_MCAST_DATA(30) |
| system-cpp-police-sys- data | WK_CPP_POLICE_SYS _DATA(10) | WK_CPU_Q_OPENFLOW (13)<br>WK_CPU_Q_CRYPTO_CONTROL(23)<br>WK_CPU_Q_EXCEPTION(24)<br>WK_CPU_Q_EGR_EXCEPTION(28)<br>WK_CPU_Q_NFL_SAMPLED_DATA(26)<br>WK_CPU_Q_GOLD_PKT(31)<br>WK_CPU_Q_RPF_FAILED(19) |
| system-cpp-police-dot1x-auth | WK_CPP_POLICE_DOT1X(11) | WK_CPU_Q_DOT1X_AUTH(0) |
| system-cpp-police- protocol-snooping | WK_CPP_POLICE_PR(12) | WK_CPU_Q_PROTO_SNOOPING(16) |

| Class Maps Names | Policer Index (Policer No.) | CPU queues (Queue No.) |
|---|---|---|
| system-cpp-police-dhcp-snooping | WK_CPP_DHCP_SNOOPING(6) | WK_CPU_Q_DHCP_SNOOPING(17) |
| system-cpp-police-sw-forward | WK_CPP_POLICE_SW_FWD (13) | WK_CPU_Q_SW_FORWARDING_Q(14)<br>WK_CPU_Q_LOGGING(21)<br>WK_CPU_Q_L2_LVX_DATA_PACK (11) |
| system-cpp-police-forus | WK_CPP_POLICE_FORUS(14) | WK_CPU_Q_FORUS_ADDR_RESOLUTION(5)<br>WK_CPU_Q_FORUS_TRAFFIC(2) |
| system-cpp-police- multicast-end-station | WK_CPP_POLICE_MULTICAST_SNOOPING(15) | WK_CPU_Q_MCAST_END_STA TION_SERVICE(20) |
| system-cpp-default | WK_CPP_POLICE_DEFAULT_POLICER(16) | WK_CPU_Q_INTER_FED_TRAFFIC(7)<br>WK_CPU_Q_EWLC_CONTROL(9)<br>WK_CPU_Q_EWLC_DATA(10) |
| system-cpp-police-stackwise-virt-control | WK_CPP_STACKWISE_VIRTUAL_CONTROL(5) | WK_CPU_Q_STACKWISE_VIRTUAL_CONTROL (29) |
| system-cpp-police-l2lvx-control | WK_CPP_ L2_LVX_CONT_PACK(4) | WK_CPU_Q_L2_LVX_CONT_PACK(8) |
| system-cpp-police-high-rate-app | WK_CPP_HIGH_RATE_APP(18) | WK_CPU_Q_HIGH_RATE_APP(23) |
| system-cpp-police-system-critical | WK_CPP_SYSTEM_CRITICAL(3) | WK_CPU_Q_SYSTEM_CRITICAL(25) |

The following table lists the CPU queues and the feature(s) associated with each CPU queue.

***Table 53: CPU Queues and Associated Features***

| CPU queues (Queue No.) | Feature(s) |
|---|---|
| WK_CPU_Q_DOT1X_AUTH(0) | IEEE 802.1x Port-Based Authentication |

| CPU queues (Queue No.) | Feature(s) |
|---|---|
| WK_CPU_Q_L2_CONTROL(1) | Dynamic Trunking Protocol (DTP)<br>VLAN Trunking Protocol (VTP)<br>Port Aggregation Protocol (PAgP)<br>Client Information Signaling Protocol (CISP)<br>Message session relay protocol<br>Multiple VLAN Registration Protocol (MVRP)<br>Metropolitan Mobile Network (MMN)<br>Link Level Discovery Protocol (LLDP)<br>UniDirectional Link Detection (UDLD)<br>Link Aggregation Control Protocol (LACP)<br>Cisco Discovery Protocol (CDP)<br>Spanning Tree Protocol (STP) |
| WK_CPU_Q_FORUS_TRAFFIC(2) | Host such as Telnet,Pingv4 and Pingv6, and SNMP<br>Keepalive / loopback detection<br>Initiate-Internet Key Exchange (IKE) protocol (IPSec) |
| WK_CPU_Q_ICMP_GEN(3) | ICMP - destination unreachable<br>ICMP-TTL expired |

| CPU queues (Queue No.) | Feature(s) |
|---|---|
| WK_CPU_Q_ROUTING_CONTROL(4) | Routing Information Protocol version 1 (RIPv1)<br>RIPv2<br>Interior Gateway Routing Protocol (IGRP)<br>Border Gateway Protocol (BGP)<br>PIM-UDP<br>Virtual Router Redundancy Protocol (VRRP)<br>Hot Standby Router Protocol version 1 (HSRPv1)<br>HSRPv2<br>Gateway Load Balancing Protocol (GLBP)<br>Label Distribution Protocol (LDP)<br>Web Cache Communication Protocol (WCCP)<br>Routing Information Protocol next generation (RIPng)<br>Open Shortest Path First (OSPF)<br>Open Shortest Path First version 3(OSPFv3)<br>Enhanced Interior Gateway Routing Protocol (EIGRP)<br>Enhanced Interior Gateway Routing Protocol version 6 (EIGRPv6)<br>DHCPv6<br>Protocol Independent Multicast (PIM)<br>Protocol Independent Multicast version 6 (PIMv6)<br>Hot Standby Router Protocol next generation (HSRPng)<br>IPv6 control<br>Generic Routing Encapsulation (GRE) keepalive<br>Network Address Translation (NAT) punt<br>Intermediate System-to-Intermediate System (IS-IS) |
| WK_CPU_Q_FORUS_ADDR_RESOLUTION(5) | Address Resolution Protocol (ARP)<br>IPv6 neighbor advertisement and neighbor solicitation |
| WK_CPU_Q_ICMP_REDIRECT(6) | Internet Control Message Protocol (ICMP) redirect |

| CPU queues (Queue No.) | Feature(s) |
|---|---|
| WK_CPU_Q_INTER_FED_TRAFFIC(7) | Layer 2 bridge domain inject for internal communication. |
| WK_CPU_Q_L2_LVX_CONT_PACK(8) | Exchange ID (XID) packet |
| WK_CPU_Q_EWLC_CONTROL(9) | Embedded Wirelss Controller (eWLC) [Control and Provisioning of Wireless Access Points (CAPWAP) (UDP 5246)] |
| WK_CPU_Q_EWLC_DATA(10) | eWLC data packet (CAPWAP DATA, UDP 5247) |
| WK_CPU_Q_L2_LVX_DATA_PACK(11) | Unknown unicast packet punted for map request. |
| WK_CPU_Q_BROADCAST(12) | All types of broadcast |
| WK_CPU_Q_OPENFLOW(13) | Learning cache overflow (Layer 2 + Layer 3) |
| WK_CPU_Q_CONTROLLER_PUNT(14) | Data - access control list (ACL) Full<br>Data - IPv4 options<br>Data - IPv6 hop-by-hop<br>Data - out-of-resources / catch all<br>Data - Reverse Path Forwarding (RPF) incomplete<br>Glean packet |
| WK_CPU_Q_TOPOLOGY_CONTROL(15) | Spanning Tree Protocol (STP)<br>Resilient Ethernet Protocol (REP)<br>Shared Spanning Tree Protocol (SSTP) |
| WK_CPU_Q_PROTO_SNOOPING(16) | Address Resolution Protocol (ARP) snooping for Dynamic ARP Inspection (DAI) |
| WK_CPU_Q_DHCP_SNOOPING(17) | DHCP snooping |
| WK_CPU_Q_TRANSIT_TRAFFIC(18) | This is used for packets punted by NAT, which need to be handled in the software path. |
| WK_CPU_Q_RPF_FAILED(19) | Data – mRPF (multicast RPF) failed |
| WK_CPU_Q_MCAST_END_STATION_SERVICE(20) | Internet Group Management Protocol (IGMP) / Multicast Listener Discovery (MLD) control |
| WK_CPU_Q_LOGGING(21) | Access control list (ACL) logging |
| WK_CPU_Q_PUNT_WEBAUTH(22) | Web Authentication |

| CPU queues (Queue No.) | Feature(s) |
|---|---|
| WK_CPU_Q_HIGH_RATE_APP(23) | Wired Application Visibility and Control (WDAVC) traffic<br>Network-Based Application Recognition (NBAR) traffic |
| WK_CPU_Q_EXCEPTION(24) | IKE indication<br>IP learning violation<br>IP port security violation<br>IP Static address violation<br>IPv6 scope check<br>Remote Copy Protocol (RCP) exception<br>Unicast RPF fail |
| WK_CPU_Q_SYSTEM_CRITICAL(25) | Media Signaling/ Wireless Proxy ARP |
| WK_CPU_Q_NFL_SAMPLED_DATA(26) | Netflow sampled data and Media Services Proxy (MSP) |
| WK_CPU_Q_LOW_LATENCY(27) | Bidirectional Forwarding Detection (BFD), Precision Time Protocol (PTP) |
| WK_CPU_Q_EGR_EXCEPTION(28) | Egress resolution exception |
| WK_CPU_Q_STACKWISE_VIRTUAL_CONTROL(29) | Front side stacking protocols, namely SVL |
| WK_CPU_Q_MCAST_DATA(30) | Data - (S,G) creation<br>Data - local joins<br>Data - PIM Registration<br>Data - SPT switchover<br>Data - Multicast |
| WK_CPU_Q_GOLD_PKT(31) | Gold |

# User-Configurable Aspects of CoPP

You can perform these tasks to manage control plane traffic:

**Note** All `system-cpp-policy` configurations must be saved so they are retained after reboot.

### Enable or Disable a Policer for CPU Queues

Enable a policer for a CPU queue, by configuring a policer action (in packets per second) under the corresponding class-map, within the `system-cpp-policy` policy-map.

Disable a policer for CPU queue, by removing the policer action under the corresponding class-map, within the `system-cpp-policy` policy-map.

**Note** If a default policer is already present, carefully consider and control its removal; otherwise the system may see a CPU hog or other anomalies, such as control packet drops.

### Change the Policer Rate

You can do this by configuring a policer rate action (in packets per second), under the corresponding class-map, within the`system-cpp-policy` policy-map.

When setting a policer rate, note that the rate you set is automatically converted to the nearest multiple of 200. For instance, if you set the policer rate of a CPU queue 100 pps, the system changes it to 200; or if set the policer rate to 650, the system changes it to 600. See *Example: Setting the Default Policer Rates for All CPU Queues* in this chapter, for sample output that displays this behavior.

### Set Policer Rates to Default

Set the policer for CPU queues to their default values, by entering the **cpp system-default** command in global configuration mode.

# Upgrading or Downgrading the Software Version

## Software Version Upgrades and CoPP

When you upgrade the software version on your device, the system checks and make the necessary updates as required for CoPP (For instance, it checks for the `system-cpp-policy` policy map and creates it if missing). You may also have to complete certain tasks before or after the upgrade activity. This is to ensure that any configuration updates are reflected correctly and CoPP continues to work as expected. Depending on the method you use to upgrade the software, upgrade-related tasks may be optional or recommended in some scenarios, and mandatory in others.

The system actions and user actions for an upgrade, are described here. Also included, are any release-specfic caveats.

### System Actions for an Upgrade

When you upgrade the software version on your device, the system performs these actions. This applies to all upgrade methods:

- If the device did not have a `system-cpp-policy` policy map before upgrade, then on upgrade, the system creates a default policy map.
- If the device had a `system-cpp-policy` policy map before upgrade, then on upgrade, the system does not re-generate the policy.

### User Actions for an Upgrade

User actions for an upgrade – depending on upgrade method:

| Upgrade Method | Condition | Action Time and Action | Purpose |
|---|---|---|---|
| Regular[13] | None | **After upgrade (required)**<br>Enter the **cpp system-default** command in global configuration mode | To get the latest, default policer rates. |
| In-Service Software Upgrade (ISSU)[14] | If there are user-defined classes in the existing software version<br>or<br>If there are system-defined classes in the existing software version that are deprecated in a later release (for example: system-cpp-police-control-low-priority). | **Before and after upgrade (required)**<br>Enter the **cpp system-default** command in global configuration mode | Enter the command before upgrade, to ensure that any required system configuration is updated, ensuring smooth ISSU operation.<br>Enter the command after upgrade for the latest, default policer rates. |

13 Refers to a software upgrade method that involves a reload of the switch. Can be install or bundle mode.
14 ISSU is supported only from one extended maintenance release to another. For more information, see In-Service Software Upgrade (ISSU).

## Software Version Downgrades and CoPP

The system actions and user actions for a downgrade, are described here.

### System Actions for a Downgrade

When you downgrade the software version on your device, the system performs these actions. This applies to all downgrade methods:

- The system retains the `system-cpp-policy` policy map on the device, and installs it on the control plane.

### User Actions for a Downgrade

User actions for a downgrade:

| Upgrade Method | Condition | Action Time and Action | Purpose |
|---|---|---|---|
| Regular[15] | None | No action required | Not applicable |
| In-Service Software Upgrade (ISSU)[16] | None | No action required | Not applicable |

15 Refers to a software upgrade method that involves a reload of the switch. Can be install or bundle mode.
16 ISSU downgrades are not supported.

If you downgrade the software version and then upgrade, the system action and user actions that apply are the same as those mentioned for upgrades.

# How to Configure CoPP

## Enabling a CPU Queue or Changing the Policer Rate

The procedure to enable a CPU queue and change the policer rate of a CPU queue is the same. Follow these steps:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **policy-map** *policy-map-name*<br>**Example:**<br>`Device(config)# policy-map system-cpp-policy`<br>`Device(config-pmap)#` | Enters the policy map configuration mode. |
| **Step 4** | **class** *class-name*<br>**Example:**<br>`Device(config-pmap)# class system-cpp-police-protocol-snooping`<br>`Device(config-pmap-c)#` | Enters the class action configuration mode. Enter the name of the class that corresponds to the CPU queue you want to enable. See table *System-Defined Values for CoPP*. |
| **Step 5** | **police rate** *rate* **pps**<br>**Example:**<br>`Device(config-pmap-c)# police rate 100 pps`<br>`Device(config-pmap-c-police)#` | Specifies an upper limit on the number of incoming packets processed per second, for the specified traffic class.<br>**Note**<br>The rate you specify is applied to all CPU queues that belong to the class-map you have specified. |
| **Step 6** | exit<br>**Example:**<br>`Device(config-pmap-c-police)# exit`<br>`Device(config-pmap-c)# exit` | Returns to the global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config-pmap)# exit`<br>`Device(config)#` | |
| **Step 7** | **control-plane**<br>**Example:**<br>`Device(config)# control-plane`<br>`Device(config-cp)#` | Enters the control plane (config-cp) configuration mode |
| **Step 8** | **service-policy input** *policy-name*<br>**Example:**<br>`Device(config)# control-plane`<br>`Device(config-cp)#service-policy input system-cpp-policy`<br>`Device(config-cp)#` | Installs system-cpp-policy in FED. This command is required for you to see the FED policy. Not configuring this command will lead to an error. |
| **Step 9** | **end**<br>**Example:**<br>`Device(config-cp)# end` | Returns to the privileged EXEC mode. |
| **Step 10** | **show policy-map control-plane**<br>**Example:**<br>`Device# show policy-map control-plane` | Displays all the classes configured under `system-cpp policy`, the rates configured for the various traffic types, and statistics |

# Disabling a CPU Queue

Follow these steps to disable a CPU queue:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **policy-map** *policy-map-name*<br>**Example:** | Enters the policy map configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(config)# policy-map system-cpp-policy`<br>`Device(config-pmap)#` | |
| **Step 4** | **class** *class-name*<br>**Example:**<br>`Device(config-pmap)# class system-cpp-police-protocol-snooping`<br>`Device(config-pmap-c)#` | Enters the class action configuration mode. Enter the name of the class that corresponds to the CPU queue you want to disable. See the table, *System-Defined Values for CoPP*. |
| **Step 5** | **no police rate** *rate* **pps**<br>**Example:**<br>`Device(config-pmap-c)# no police rate 100 pps` | Disables incoming packet processing for the specified traffic class.<br>**Note**<br>This disables all CPU queues that belong to the class-map you have specified. |
| **Step 6** | **end**<br>**Example:**<br>`Device(config-pmap-c)# end` | Returns to the privileged EXEC mode. |
| **Step 7** | **show policy-map control-plane**<br>**Example:**<br>`Device# show policy-map control-plane` | Displays all the classes configured under `system-cpp policy` and the rates configured for the various traffic types and statistics. |

# Setting the Default Policer Rates for All CPU Queues

Follow these steps to set the policer rates for all CPU queues to their default rates:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **cpp system-default**<br>**Example:**<br>`Device(config)# cpp system-default`<br>`Defaulting CPP : Policer rate for all classes will be set to their defaults` | Sets the policer rates for all the classes to the default rate. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Returns to the privileged EXEC mode. |
| **Step 5** | **show platform hardware fed switch** {*switch-number* \| *active* \| *standby*} **qos que stats internal cpu policer**<br>**Example:**<br>`Device# show platform hardware fed switch 1 qos que stat internal cpu policer` | Displays the rates configured for the various traffic types. |

# Configuration Examples for CoPP

## Example: Enabling a CPU Queue or Changing the Policer Rate of a CPU Queue

This example shows how to enable a CPU queue or to change the policer rate of a CPU queue. Here the **class system-cpp-police-protocol-snooping** CPU queue is enabled with the policer rate of **2000 pps** .

```
Device> enable
Device# configure terminal
Device(config)# policy-map system-cpp-policy
Device(config-pmap)# class system-cpp-police-protocol-snooping
Device(config-pmap-c)# police rate 2000 pps
Device(config-pmap-c-police)# end


Device# show policy-map control-plane

Control Plane

  Service-policy input: system-cpp-policy

    <output truncated>


    Class-map: system-cpp-police-dot1x-auth (match-any)
      0 packets, 0 bytes
```

```
    5 minute offered rate 0000 bps, drop rate 0000 bps
    Match: none
    police:
        rate 1000 pps, burst 244 packets
      conformed 0 bytes; actions:
        transmit
      exceeded 0 bytes; actions:
        drop

  Class-map: system-cpp-police-protocol-snooping (match-any)
    0 packets, 0 bytes
    5 minute offered rate 0000 bps, drop rate 0000 bps
    Match: none
    police:
        rate 2000 pps, burst 488 packets
      conformed 0 bytes; actions:
        transmit
      exceeded 0 bytes; actions:
        drop

  <output truncated>

  Class-map: class-default (match-any)
    0 packets, 0 bytes
    5 minute offered rate 0000 bps, drop rate 0000 bps
    Match: any
```

# Example: Disabling a CPU Queue

This example shows how to disable a CPU queue. Here the **class system-cpp-police-protocol-snooping** CPU queue is disabled.

```
Device> enable
Device# configure terminal
Device(config)# policy-map system-cpp-policy
Device(config-pmap)# class system-cpp-police-protocol-snooping
Device(config-pmap-c)# no police rate 100 pps
Device(config-pmap-c)# end


Device# show running-config | begin system-cpp-policy

policy-map system-cpp-policy
 class system-cpp-police-data
  police rate 200 pps
 class system-cpp-police-sys-data
  police rate 100 pps
 class system-cpp-police-sw-forward
  police rate 1000 pps
 class system-cpp-police-multicast
  police rate 500 pps
 class system-cpp-police-multicast-end-station
  police rate 2000 pps
 class system-cpp-police-punt-webauth
 class system-cpp-police-l2-control
 class system-cpp-police-routing-control
  police rate 500 pps
 class system-cpp-police-control-low-priority
 class system-cpp-police-wireless-priority1
 class system-cpp-police-wireless-priority2
```

```
 class system-cpp-police-wireless-priority3-4-5
 class system-cpp-police-topology-control
 class system-cpp-police-dot1x-auth
 class system-cpp-police-protocol-snooping
 class system-cpp-police-forus
 class system-cpp-default

<output truncated>
```

# Example: Setting the Default Policer Rates for All CPU Queues

This example shows how to set the policer rates for all CPU queues to their default and then verify the setting.

**Note** For some CPU queues, the `default rate` and the `set rate` values will not be the same, even if you set the default rate for all classes. This because the set rate is rounded off to the nearest multiple of 200. This behavior is controlled by the clock speed of your device. In the sample output below, the default and set rate values for `DHCP Snooping` and `NFL SAMPLED DATA` display this difference.

```
Device> enable
Device# configure terminal
Device(config)# cpp system-default
Defaulting CPP : Policer rate for all classes will be set to their defaults
Device(config)# end

Device# show platform hardware fed switch 1 qos queue stats internal cpu policer

CPU Queue Statistics
============================================================================================

                                                (default) (set)     Queue        Queue
QId PlcIdx  Queue Name               Enabled     Rate     Rate     Drop(Bytes)  Drop(Frames)
--------------------------------------------------------------------------------------------
0    11     DOT1X Auth                 Yes       1000     1000     0            0

1    1      L2 Control                 Yes       2000     2000     0            0

2    14     Forus traffic              Yes       1000     1000     0            0

3    0      ICMP GEN                   Yes       600      600      0            0

4    2      Routing Control            Yes       5400     5400     0            0

5    14     Forus Address resolution   Yes       4000     4000     0            0

6    0      ICMP Redirect              Yes       600      600      0            0

7    16     Inter FED Traffic          Yes       2000     2000     0            0

8    4      L2 LVX Cont Pack           Yes       1000     1000     0            0

9    16     EWLC Control               Yes       2000     2000     0            0

10   16     EWLC Data                  Yes       2000     2000     0            0

11   13     L2 LVX Data Pack           Yes       1000     1000     0            0

12   0      BROADCAST                  Yes       600      600      0            0

13   10     Openflow                   Yes       100      200      0            0
```

```
14   13     Sw forwarding              Yes     1000      1000     0            0

15   8      Topology Control           Yes     13000     13000    0            0

16   12     Proto Snooping             Yes     2000      2000     0            0

17   6      DHCP Snooping              Yes     500       400      0            0

18   9      Transit Traffic            Yes     500       400      0            0

19   10     RPF Failed                 Yes     100       200      0            0

20   15     MCAST END STATION          Yes     2000      2000     0            0

21   13     LOGGING                    Yes     1000      1000     0            0

22   7      Punt Webauth               Yes     1000      1000     0            0

23   18     High Rate App              Yes     13000     13000    0            0

24   10     Exception                  Yes     100       200      0            0

25   3      System Critical            Yes     1000      1000     0            0

26   10     NFL SAMPLED DATA           Yes     100       200      0            0

27   2      Low Latency                Yes     5400      5400     0            0

28   10     EGR Exception              Yes     100       200      0            0

29   5      Stackwise Virtual OOB      Yes     8000      8000     0            0

30   9      MCAST Data                 Yes     500       400      0            0

31   10     Gold Pkt                   Yes     100       200      0            0

* NOTE: CPU queue policer rates are configured to the closest hardware supported value

                      CPU Queue Policer Statistics
======================================================================
Policer     Policer Accept   Policer Accept  Policer Drop  Policer Drop
  Index          Bytes           Frames         Bytes           Frames
----------------------------------------------------------------------
0           0                0               0             0
1           0                0               0             0
2           0                0               0             0
3           0                0               0             0
4           0                0               0             0
5           0                0               0             0
6           0                0               0             0
7           0                0               0             0
8           0                0               0             0
9           0                0               0             0
10          0                0               0             0
11          0                0               0             0
12          0                0               0             0
13          0                0               0             0
14          0                0               0             0
15          0                0               0             0
16          0                0               0             0
17          0                0               0             0
18          0                0               0             0
```

```
                  Second Level Policer Statistics
==================================================================
20         52772252         688073          0             0
21         0                0               0             0

Policer Index Mapping and Settings
------------------------------------------------------------------
level-2   :   level-1                      (default)   (set)
PlcIndex  :   PlcIndex                       rate      rate
------------------------------------------------------------------
20        :   1  2  8                        13000     13000
21        :   0 4 7 9 10 11 12 13 14 15      6000      6000
==================================================================

               Second Level Policer Config
==================================================================
    level-1 level-2                              level-2
QId PlcIdx  PlcIdx  Queue Name                   Enabled
------------------------------------------------------------------
0    11      21      DOT1X Auth                   Yes
1    1       20      L2 Control                   Yes
2    14      21      Forus traffic                Yes
3    0       21      ICMP GEN                     Yes
4    2       20      Routing Control              Yes
5    14      21      Forus Address resolution     Yes
6    0       21      ICMP Redirect                Yes
7    16      -       Inter FED Traffic            No
8    4       21      L2 LVX Cont Pack             Yes
9    19      -       EWLC Control                 No
10   16      -       EWLC Data                    No
11   13      21      L2 LVX Data Pack             Yes
12   0       21      BROADCAST                    Yes
13   10      21      Openflow                     Yes
14   13      21      Sw forwarding                Yes
15   8       20      Topology Control             Yes
16   12      21      Proto Snooping               Yes
17   6       -       DHCP Snooping                No
18   13      21      Transit Traffic              Yes
19   10      21      RPF Failed                   Yes
20   15      21      MCAST END STATION            Yes
21   13      21      LOGGING                      Yes
22   7       21      Punt Webauth                 Yes
23   18      -       High Rate App                No
24   10      21      Exception                    Yes
25   3       -       System Critical              No
26   10      21      NFL SAMPLED DATA             Yes
27   2       20      Low Latency                  Yes
28   10      21      EGR Exception                Yes
29   5       -       Stackwise Virtual OOB        No
30   9       21      MCAST Data                   Yes
31   3       -       Gold Pkt                     No

                        CPP Classes to queue map
==========================================================================================
PlcIdx CPP Class                                 :  Queues
------------------------------------------------------------------------------------------
0      system-cpp-police-data                    :  ICMP GEN/BROADCAST/ICMP Redirect/
10     system-cpp-police-sys-data                :  Openflow/Exception/EGR Exception/NFL
SAMPLED DATA/Gold Pkt/RPF Failed/
13     system-cpp-police-sw-forward              :  Sw forwarding/LOGGING/L2 LVX Data Pack/
9      system-cpp-police-multicast               :  Transit Traffic/MCAST Data/
15     system-cpp-police-multicast-end-station   :  MCAST END STATION /
7      system-cpp-police-punt-webauth            :  Punt Webauth/
```

```
1      system-cpp-police-l2-control             :  L2 Control/
2      system-cpp-police-routing-control        :  Routing Control/Low Latency/
3      system-cpp-police-system-critical        :  System Critical/
4      system-cpp-police-l2lvx-control          :  L2 LVX Cont Pack/
8      system-cpp-police-topology-control       :  Topology Control/
11     system-cpp-police-dot1x-auth             :  DOT1X Auth/
12     system-cpp-police-protocol-snooping      :  Proto Snooping/
6      system-cpp-police-dhcp-snooping          :  DHCP Snooping/
14     system-cpp-police-forus                  :  Forus Address resolution/Forus traffic/
5      system-cpp-police-stackwise-virt-control :  Stackwise Virtual OOB/
16     system-cpp-default                       :  Inter FED Traffic/ EWLC Data/
18     system-cpp-police-high-rate-app          :  High Rate App/
19     system-cpp-police-ewlc-control           :  EWLC Control/
20     system-cpp-police-ios-routing            :  L2 Control/ Topology Control/ Routing
Control/ Low Latency/
21     system-cpp-police-ios-feature            :  ICMP GEN/ BROADCAST/ ICMP Redirect/ L2
LVX Cont Pack/ Proto Snooping/ Punt Webauth/ MCAST Data/ Transit Traffic/ DOT1X Auth/ Sw
forwarding/ LOGGING/ L2 LVX Data Pack/ Forus traffic/ Forus Address resolution/ MCAST END
STATION / Openflow/ Exception/ EGR Exception/ NFL SAMPLED DATA/ RPF Failed/
```

# Monitoring CoPP

Use these commands to display policer settings, such as, traffic types and policer rates (user-configured and default rates) for CPU queues:

| Command | Purpose |
|---|---|
| **show policy-map control-plane** | Displays the rates configured for the various traffic types |
| **show policy-map system-cpp-policy** | Displays all the classes configured under system-cpp policy, and policer rates |
| **show platform hardware fed switch** { *switch-number* \| *active* \| *standby* } **qos que stats internal cpu policer** | Displays the rates configured for the various traffic types |
| **show platform software fed** { *switch-number* \| *active* \| *standby* } **qos policy target status** | Displays information about policy status and the target port type. |

# Feature History for CoPP

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Control Plane Policing (CoPP) or CPP | The CoPP feature improves security on your device by protecting the CPU from unnecessary traffic, or DoS traffic, and by prioritizing control plane and management traffic.<br><br>The feature provides CLI configuration options to enable and disable CPU queues, to change the policer rate, set policer rates to default, and to create user-defined class-maps (with filters) and add them to policy map `system-cpp-policy`. |
| Cisco IOS XE Everest 16.6.1 | Changes in system-defined values for CoPP | These new system-defined classes were introduced:<br>• system-cpp-police-stackwise-virt-control<br>• system-cpp-police-l2lvx-control<br><br>These new CPU queues were added to the existing `system-cpp-default` class:<br>• WK_CPU_Q_UNUSED (7)<br>• WK_CPU_Q_EWLC_CONTROL(9)<br>• WK_CPU_Q_EWLC_DATA(10)<br><br>CPU queue WK_CPU_Q_L2_LVX_DATA_PACK (11) was added to class `system-cpp-police-sw-forward`.<br><br>CPU queue WK_CPU_Q_SGT_CACHE_FULL(27) is no longer available. |
| Cisco IOS XE Everest 16.6.4 | Change in the system behavior for policer rates that are set. | For some CPU queues, the default rate and the set rate values will not be the same, even if you set the default rate for all classes. This is because the set rate is rounded off to the nearest multiple of 200. |

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Fuji 16.8.1a | Removal of support for user-defined class-maps and changes in system-defined values for CoPP | • Starting from this release, the creation of user-defined class-maps is not supported.<br>• This new system-defined class was introduced: system-cpp-police-dhcp-snooping<br>• This new CPU queue was added to the existing system-cpp-default class: WK_CPU_Q_INTER_FED_TRAFFIC<br>• These CPU queues are no longer available:<br>  • WK_CPU_Q_SHOW_FORWARD<br>  • WK_CPU_Q_UNUSED<br>• The default policer rate (pps) for some CPU queues has changed:<br>  • The default rate for WK_CPU_Q_EXCEPTION(24) was changed to 100<br>  • The default rate for all the CPU queues under system-cpp-default was increased to 2000.<br>  • The default rate for all the CPU queues under system-cpp-police-forus was increased to 4000. |
| | Control Plane Policing (CoPP) or CPP introduced on the High Performance models in the series | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches.<br>All Cisco IOS XE Fuji 16.8.1a release changes apply to all models in the series. |
| Cisco IOS XE Fuji 16.9.1 | Changes in system-defined values for CoPP | Starting with this release, eighteen system-defined classes are created under `system-cpp-policy`.<br>These new system-defined classes were introduced:<br>• system-cpp-police-high-rate-app<br>• system-cpp-police-system-critical<br>CPU queue WK_CPU_Q_OPENFLOW (13) was added to class `system-cpp-police-sys-data`.<br>CPU queue WK_CPU_Q_LEARNING_CACHE_OVFL(13) is no longer available. |

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Fuji 16.9.4 | Deprecation of system-defined class map | This system-defined class map was deprecated: system-cpp-police-control-low-priority |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 45

# Configuring Lawful Intercept


- Prerequisites for Lawful Intercept, on page 777
- Restrictions for Lawful Intercept, on page 777
- Information About Lawful Intercept, on page 778
- How to Configure Lawful Intercept, on page 785
- Configuration Examples for Lawful Intercept, on page 788
- Feature History for Lawful Intercept, on page 789


## Prerequisites for Lawful Intercept

- You must be running images that support secure shell (SSH). Lawful intercept is not supported on images that do not support SSH.
- The switch you are logged into must have the highest access level (L15). To log in with L15 access, enter the **enable** command and specify the highest-level password defined for the switch.
- The time of day on the switch and the mediation device (supplied by a third-party vendor) connected to the network must be must be synchronized; also, configure Network Time Protocol (NTP) on both the switch and the mediation device.
- (Optional) It might be helpful to use a loopback interface for the switch to communicate with the mediation device. If you do not use a loopback interface, you must configure the mediation device with multiple physical interfaces on the switch to handle network failures.

## Restrictions for Lawful Intercept

- This feature is not supported on the C9500-12Q, C9500-16X, C9500-24Q, C9500-40X models of the Cisco Catalyst 9500 Series Switches.
- Intercept or tap can be configured using only SNMP. Also, configuring interface-specific intercept is not allowed.
- The CISCO-IP-TAP-MIB does not support the virtual routing and forwarding (VRF) OID citapStreamVRF.
- IPv4 multicast, IPv6 unicast, and IPv6 multicast flows are not supported. Only IPv4 unicast traffic is supported.

- Lawful Intercept is not supported on Layer 2 interfaces. However, lawful intercept can intercept traffic on VLANs that run over a Layer 2 interface.
- Lawful Intercept is not supported in packets that are encapsulated within other packets, for example, tunneled packets or Q-in-Q packets.
- Lawful Intercept is not supported for Layer 2 taps.
- Lawful Intercept is not supported in packets that are subject to Layer 3 or Layer 4 rewrite, for example, Network Address Translation (NAT) or TCP reflexive.

# Information About Lawful Intercept

## Overview of Lawful Intercept

Lawful intercept is a process that enables a law enforcement agency to perform electronic surveillance on an individual (a target) as authorized by a judicial (court) or administrative order. To facilitate the lawful intercept process, certain legislation and regulations require service providers and Internet Service Providers (ISPs) to implement their networks to explicitly support authorized electronic surveillance.

Surveillance is performed through the use of wiretaps on traditional telecommunications and internet services in voice, data, and multiservice networks. The law enforcement agencies deliver a request for a wiretap to the target's service provider who is responsible for intercepting data communication to and from the individual. The service provider uses the target's IP address to determine which of its edge devices handles the target's traffic (data communication). The service provider then intercepts the target's traffic as it passes through the device, and sends a copy of the intercepted traffic to the LEA without the target's knowledge.

The Lawful Intercept feature supports the Communications Assistance for Law Enforcement Act (CALEA), which describes how service providers in the United States must support lawful intercept. Currently, lawful intercept is defined by the following standards:

- Telephone Industry Association (TIA) specification J-STD-025
- Packet Cable Electronic Surveillance Specification (PKT-SP-ESP-101-991229)

For information about the Cisco lawful intercept solution, contact your Cisco account representative.

**Note** The Lawful Intercept feature supports the interception of IPv4 protocol as defined by the citapStreamprotocol object in the CISCO-IP-TAB-MIB that includes voice and date interception.

## Benefits of Lawful Intercept

- Allows multiple LEAs to run a lawful intercept on the same target without each other's knowledge.
- Does not affect subscriber services on the device.
- Supports wiretaps in both the input and output direction.
- Supports wiretaps of Layer 1 and Layer 3 traffic. Layer 2 traffic is supported as IP traffic over VLANs.

- Supports Layer 3 physical interfaces or Switch Virtual Interfaces (SVI).
- Supports wiretaps of individual subscribers that share a single physical interface.
- Cannot be detected by the target. Neither the network administrator nor the calling party is aware that packets are being copied or that the call is being tapped.
- Uses Simple Network Management Protocol Version 3 (SNMPv3) and security features such as the View-based Access Control Model (SNMP-VACM-MIB) and the User-based Security Model (SNMP-USM-MIB) to restrict access to lawful intercept information and components.
- Hides information about lawful intercepts from all but the most privileged users. An administrator must set up access rights to enable privileged users to access lawful intercept information.
- Provides two secure interfaces for performing an intercept; one for setting up the wiretap and one for sending the intercepted traffic to the LEA.

# CALEA for Voice

The Communications Assistance for Law Enforcement Act (CALEA) for Voice feature allows the lawful interception of voice conversations that are running on VoIP. Although the devices are not voice gateway devices, VoIP packets traverse the devices at the edge of the service provider network.

When an approved government agency determines that a telephone conversation is interesting, CALEA for Voice copies the IP packets comprising the conversation and sends the duplicate packets to the appropriate monitoring device for further analysis.

# Configuration Guidelines

- To deploy Lawful Intercept at a node, do not configure optimized ACL logging, VLAN access control list (VACL) capture, or Intrusion Detection System (IDS) at the node. Deploying lawful intercept at the node causes unpredictable behavior in optimized ACL logging, VACL capture, and IDS.
- The maximum number of TAPs supported on Catalyst 9500 series switches is 160.
- When provisioning the mediation device, if the interface index passed is zero, the switch selects the best possible interface to reach the mediation device. If the interface index is set to another value, the switch uses that interface index to reach the mediation device.
- (Optional) The domain name for both the device and the mediation device can be registered in the Domain Name System (DNS).
- The mediation device must have an access function.
- You must add the mediation device to the SNMP user group that has access to the CISCO-TAP2-MIB view. Specify the username of the mediation device as the user to add to the group.
- When you add the mediation device as a CISCO-TAP2-MIB user, you must also include the mediation device’s authorization password.
- The device intercepts and replicates packets even if the packets are later dropped, for example, due to rate limiting or an access control list (ACL) deny statement.
- Lawful intercept ACLs are applied internally to both the ingress and the egress directions of an interface.

- Packets that are subject to hardware rate limiting are processed by lawful intercept as follows:
  - Packets that are dropped by the rate limiter are not intercepted or processed.
  - Packets that are passed by the rate limiter are intercepted and processed.
- If multiple law enforcement agencies use a single mediation device and each of these agencies is executing a wiretap on the same target, the device sends a single packet to the mediation device. It is up to the mediation device to duplicate the packet for each law enforcement agency.
- Lawful intercept can intercept IPv4 packets with values that match a combination of one or more of the following fields:
  - Destination IP address and mask
  - Destination port range
  - Source IP address and mask
  - Source port range
  - Protocol ID

# Network Components Used for Lawful Intercept

This section describes the network components used for lawful intercept.

### Mediation Device

A mediation device (supplied by a third-party vendor) handles most of the processing for lawful intercept. The mediation device:

- Provides the interface used to set up and provision the lawful intercept.
- Generates requests to other network devices to set up and run the lawful intercept.
- Converts the intercepted traffic into the format required by the LEA (which can vary from country to country) and sends a copy of the intercepted traffic to the LEA without the target's knowledge.

**Note** If multiple LEAs are performing intercepts on the same target, the mediation device must make a copy of the intercepted traffic for each LEA. The mediation device is also responsible for restarting the lawful intercepts that are disrupted due to a failure.

### Lawful Intercept Administration

Lawful intercept administration (LIA) provides the authentication interface for lawful intercept or wiretap requests and administration.

### Intercept Access Point

An intercept access point (IAP) is a device that provides information to lawful intercept. There are two types of IAPs:

- Identification (ID) IAP—A device, such as an authentication, authorization, and accounting (AAA) server that provides intercept-related information (IRI) for the intercept (for example, the target's username and system IP address) or call agents for VoIP. The IRI helps a service provider determine which content IAP (switch) the target's traffic passes through.
- Content IAP—A device that the target's traffic passes through. The content IAP:
  - Intercepts traffic to and from the target for the length of time specified in the court order. The device continues to forward traffic to its destination to ensure that the wiretap is undetected.
  - Creates a copy of the intercepted traffic, encapsulates it in User Datagram Protocol (UDP) packets, and forwards the packets to the mediation device without the target's knowledge. Note that the IP option header is not supported.

**Note** If multiple LEAs are performing intercepts on the same target, the mediation device must make a copy of the intercepted traffic for each LEA.

### Content Intercept Access Point

Content IAP intercepts the interested data stream, duplicates the content, and sends the duplicated content to the mediation device. The mediation device receives the data from the ID IAP and content IAP, converts the information to the required format depending on country-specific requirements and forwards it to the law enforcement agency.

## Lawful Intercept Processing

After acquiring a court order or warrant to perform surveillance, the LEA delivers a surveillance request to the target's service provider. Service provider personnel use an administration function that runs on the mediation device to configure a lawful intercept to monitor the target's electronic traffic for a specific period of time (as defined in the court order).

After the intercept is configured, user intervention is no longer required. The administration function communicates with other network devices to set up and execute the lawful intercept. The following sequence of events occurs during a lawful intercept:

1. The administration function contacts the ID IAP for intercept-related information (IRI), such as the target's username and the IP address of the system, to determine which content IAP (switch) the target's traffic passes through.
2. After identifying the device that handles the target's traffic, the administration function sends SNMPv3 get and set requests to the device's Management Information Base (MIB) to set up and activate the lawful intercept. The CISCO-TAP2-MIB is the supported lawful intercept MIB to provide per-subscriber intercepts.
3. During lawful intercept, the device:
   a. Examines incoming and outgoing traffic and intercepts any traffic that matches the specifications of the lawful intercept request.
   b. Creates a copy of the intercepted traffic and forwards the original traffic to its destination so that the target does not suspect anything.

c. Encapsulates the intercepted traffic in UDP packets, and forwards the packets to the mediation device without the target's knowledge.

**Note** The process of intercepting and duplicating the target's traffic does not add detectable latency in the traffic stream.

4. The mediation device converts the intercepted traffic into the required format and sends it to a collection function running at the LEA. Here, the intercepted traffic is stored and processed.

**Note** If the device intercepts traffic that is not allowed by the judicial order, the mediation device filters out the excess traffic and sends only the traffic allowed by the judicial order to the LEA.

5. When the lawful intercept expires, the device stops intercepting the target's traffic.

# Lawful Intercept MIBs

- CISCO-TAP2-MIB: Used for lawful intercept processing.
- CISCO-IP-TAP-MIB: Used for intercepting Layer 3 (IPv4) traffic.

Due to its sensitive nature, the Cisco lawful intercept MIBs are only available in software images that support the Lawful Intercept feature. To access the Cisco IOS MIB Locator page, go to:

http://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index.

### CISCO-TAP2-MIB

The CISCO-TAP2-MIB contains SNMP management objects that control lawful intercepts. The mediation device uses the MIB to configure and run lawful intercepts on targets whose traffic passes through the device.

The CISCO-TAP2-MIB contains several tables that provide information for lawful intercepts that are running on the device:

- cTap2MediationTable: Contains information about each mediation device that is currently running lawful intercept on the device. Each table entry provides information that the device uses to communicate with the mediation device, for example, the device's address, the interfaces to send intercepted traffic over, and the protocol to use to transmit the intercepted traffic.
- cTap2StreamTable: Contains information used to identify the traffic to intercept. Each table entry contains a pointer to a filter that is used to identify the traffic stream associated with the target of a lawful intercept. Traffic that matches the filter is intercepted, copied, and sent to the corresponding mediation device application (cTap2MediationContentId).

  The cTap2StreamTable table also contains counts of the number of packets that were intercepted, and counts of dropped packets that should have been intercepted, but were not.
- cTap2DebugTable: Contains debug information for troubleshooting lawful intercept errors.

The CISCO-TAP2-MIB also contains several SNMP notifications for lawful intercept events. For detailed descriptions of MIB objects, see corresponding MIBs.

### CISCO-TAP2-MIB Processing

The administration function (running on the mediation device) issues SNMPv3 set and get requests to the device's CISCO-TAP2-MIB to set up and initiate a lawful intercept. To do this, the administration function performs the following actions:

1. Creates a cTap2MediationTable entry to define how the device is to communicate with the mediation device executing the intercept.

**Note** The cTap2MediationNewIndex object provides a unique index for the mediation table entry.

2. Creates an entry in the cTap2StreamTable to identify the traffic stream to intercept.
3. Sets cTap2StreamInterceptEnable to true(1) to start the intercept. The device intercepts traffic in the stream until the intercept expires (cTap2MediationTimeout).

### CISCO-IP-TAP-MIB

The CISCO-IP-TAP-MIB contains the SNMP management objects to configure and execute lawful intercepts on IPv4 traffic streams that flow through the device. This MIB is an extension of the CISCO-TAP2-MIB.

You can use the CISCO-IP-TAP-MIB to configure lawful intercept on a device to intercept IPv4 packets with values that match a combination of one or more of the following fields:

- Destination IP address and mask
- Destination port range
- Source IP address and mask
- Source port range
- Protocol ID

### CISCO-IP-TAP-MIB Processing

When data is intercepted, two streams are created. One stream is for packets that originate from the target IP address to any other IP address using any port. The second stream is created for packets that are routed to the target IP address from any other address using any port. For VoIP, two streams are created, one for RTP packets from the target and the second stream for the RTP packets to target using the specific source and destination IP addresses and ports specified in the SDP information used to set up the RTP stream.

## MIB Guidelines

The following Cisco MIBs are used for lawful intercept processing. Include these MIBs in the SNMP view of lawful intercept MIBs to enable the mediation device to configure and execute wiretaps on traffic that flows through the device.

- CISCO-TAP2-MIB: Required for both types of lawful intercepts: regular and broadband.

- CISCO-IP-TAP-MIB: Required for wiretaps on Layer 3 (IPv4) streams. Supported for regular and broadband lawful intercept.
- The CISCO-IP-TAB-MIB imposes limitations on the following features:
  - If one or all of the following features are configured and functioning and lawful intercept is enabled, lawful intercept takes precedence, and the feature behaves as follows:
    - Optimized ACL logging (OAL): Does not function.
    - VLAN access control list (VACL) capturing: Does not function properly.
    - Intrusion detection system (IDS): Does not function properly.

    These features start to function after you disable or unconfigure lawful intercept.
  - IDS cannot capture traffic on its own, but captures traffic that has been intercepted by lawful intercept.

# Security Considerations

- SNMP notifications for lawful intercept must be sent to UDP port 161 on the mediation device, not port 162 (which is the SNMP default).
- The only users who should be allowed to access the lawful intercept MIBs are the mediation device and system administrators who need to know about lawful intercepts on the device. In addition, these users must have authPriv or authNoPriv access rights to access the lawful intercept MIBs. Users with NoAuthNoPriv access cannot access the lawful intercept MIBs.
- You cannot use the SNMP-VACM-MIB to create a view that includes the lawful intercept MIBs.
- The default SNMP view excludes the following MIBs:
  - CISCO-TAP2-MIB
  - CISCO-IP-TAP-MIB
  - SNMP-COMMUNITY-MIB
  - SNMP-USM-MIB
  - SNMP-VACM-MIB

We recommend that you also see the information provided in Restrictions for Lawful Intercept, on page 777 and Prerequisites for Lawful Intercept, on page 777.

# Restricting Access to the Lawful Intercept MIBs

Only the mediation device and users who need to know about lawful intercepts should be allowed to access the lawful intercept MIBs. To restrict access to these MIBs, you must:

1. Create a view that includes the Cisco lawful intercept MIBs.
2. Create an SNMP user group that has read-and-write access to the view. Only users assigned to this user group can access information in the MIBs.

3. Add users to the Cisco lawful intercept user groups to define who can access the MIBs along with information, if any, related to lawful intercepts. Be sure to add the mediation device as a user in this group; otherwise, the device cannot perform lawful intercepts.

**Note** Access to the Cisco lawful intercept MIB view should be restricted to the mediation device and to system administrators who need to be aware of lawful intercepts on the device. To access the MIB, users must have level-15 access rights on the device.

# How to Configure Lawful Intercept

## Creating a Restricted SNMP View of Lawful Intercept MIBs

To create and assign users to an SNMP view that includes the Cisco lawful intercept MIBs, perform the steps provided in this section.

### Before you begin

- SNMPv3 must be configured on the device.

**Note** Issue the commands in global configuration mode with level-15 access rights.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode. Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **snmp-server view** *view-name MIB-name* **included**<br>**Example:**<br>`Device(config)# snmp-server view exampleView ciscoTap2MIB included` | Creates an SNMP view that includes the CISCO-TAP2-MIB, where *exampleView* is the name of the view to create for the MIB. This MIB is required for both regular and broadband lawful intercept. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **snmp-server view** *view-name MIB-name* **included**<br>**Example:**<br>`Device(config)# snmp-server view exampleView ciscoIpTapMIB included` | Adds the CISCO-IP-TAP-MIB to the SNMP view. |
| **Step 5** | **snmp-server view** *view-name MIB-name* **included**<br>**Example:**<br>`Device(config)# snmp-server view exampleView cisco802TapMIB included` | Adds the CISCO-802-TAP-MIB to the SNMP view. |
| **Step 6** | **snmp-server view** *view-name MIB-name* **included**<br>**Example:**<br>`Device(config)# snmp-server view exampleView ciscoUserConnectionTapMIB included` | Adds the CISCO-USER-CONNECTION-TAP-MIB to the SNMP view. |
| **Step 7** | **snmp-server view** *view-name MIB-name* **included**<br>**Example:**<br>`Device(config)# snmp-server view exampleView ciscoMobilityTapMIB included` | Adds the CISCO-MOBILITY-TAP-MIB to the SNMP view. |
| **Step 8** | **snmp-server group** *group-name* **v3 auth read** *view-name* **write** *view-name*<br>**Example:**<br>`Device(config)# snmp-server group exampleGroup v3 auth read exampleView write exampleView` | Creates an SNMP user group that has access to the LI MIB view and defines the group's access rights to the view. |
| **Step 9** | **snmp-server user** *user-name group-name* **v3 auth md5** *auth-password*<br>**Example:**<br>`Device(config)# snmp-server user exampleUser exampleGroup v3 auth md5 examplePassword` | Adds users to the specified user group. |
| **Step 10** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Enabling SNMP Notifications for Lawful Intercept

SNMP automatically generates notifications for lawful intercept events. To configure the device to send lawful intercept notifications to the mediation device, perform the steps in this section.

### Before you begin

- SNMPv3 must be configured on the device..

**Note** Issue the commands in global configuration mode with level-15 access rights.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **snmp-server host** *ip-address* **community-string udp-port** *port notification-type*<br>**Example:**<br>`Device(config)# snmp-server host 10.2.2.1 community-string udp-port 161 udp` | Specifies the IP address of the mediation device and the password-like community-string that is sent with a notification request.<br>• For lawful intercept, the **udp-port** must be 161 and not 162 (the SNMP default). |
| **Step 4** | **snmp-server enable traps snmp authentication linkup linkdown coldstart warmstart**<br>**Example:**<br>`Device(config)# snmp-server enable traps snmp authentication linkup linkdown coldstart warmstart` | Configures the device to send RFC 1157 notifications to the mediation device.<br>• These notifications indicate authentication failures, link status (up or down), and device restarts. |
| **Step 5** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Disabling SNMP Notifications

To disable SNMP notifications on the device, perform the steps provided in this section.

**Note** To disable lawful intercept notifications, use SNMPv3 to set the CISCO-TAP2-MIB object cTap2MediationNotificationEnable to false(2). To re-enable lawful intercept notifications through SNMPv3, reset the object to true(1).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode. Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **no snmp-server enable traps**<br>**Example:**<br>`Device(config)# no snmp-server enable traps` | Disables all SNMP notification types that are available on your system. |
| **Step 4** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuration Examples for Lawful Intercept

## Example: Enabling Mediation Device Access Lawful Intercept MIBs

The following example shows how to enable the mediation device to access the lawful intercept MIBs. It creates an SNMP view (tapV) that includes three LI MIBs (CISCO-TAP2-MIB, CISCO-IP-TAP-MIB, CISCO-802-TAP-MIB). It also creates a user group that has read, write, and notify access to MIBs in the tapV view.

```
Device> enable
Device# configure terminal
Device(config)# snmp-server view tapV ciscoTap2MIB included
Device(config)# snmp-server view tapV ciscoIpTapMIB included
```

```
Device(config)# snmp-server view tapV cisco802TapMIB included
Device(config)# snmp-server group tapGrp v3 auth read tapV write tapV notify tapV
Device(config)# snmp-server user MDuser tapGrp v3 auth md5 MDpasswd
Device(config)# snmp-server engineID local 1234
Device(config)# end
```

# Feature History for Lawful Intercept

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Fuji 16.8.1a | Lawful Intercept | The Lawful Intercept feature supports service providers in meeting the requirements of law enforcement agencies to provide electronic surveillance as authorized by a judicial or administrative order.<br><br>This feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com.

CHAPTER 46

# Configuring Authorization and Revocation of Certificates in a PKI

• Configuring Authorization and Revocation of Certificates in a PKI, on page 791

# Configuring Authorization and Revocation of Certificates in a PKI

This module describes the authorization and revocation of certificates in a public key infrastructure (PKI).

## Prerequisites for Authorization and Revocation of Certificates

### Plan Your PKI Strategy

**Tip** It is strongly recommended that you plan your entire PKI strategy before you begin to deploy actual certificates.

Authorization and revocation can occur only after you or a network administrator have completed the following tasks:

- Configured the certificate authority (CA).
- Enrolled peer devices with the CA.
- Identified and configured the protocol (such as IPsec or secure socket layer [SSL]) that is to be used for peer-to-peer communication.

You should decide which authorization and revocation strategy you are going to configure before enrolling peer devices because the peer device certificates might have to contain authorization and revocation-specific information.

### High Availability

For high availability, IPsec-secured Stream Control Transmission Protocol (SCTP) must be configured on both the active and the standby devices. For synchronization to work, the redundancy mode on the certificate servers must be set to ACTIVE/STANDBY after you configure SCTP.

# Restrictions for Authorization and Revocation of Certificates

- Depending on your Cisco IOS XE release, Lightweight Directory Access Protocol (LDAP) is supported.

# Information About Authorization and Revocation of Certificates

## PKI Authorization

PKI authentication does not provide authorization. Current solutions for authorization are specific to the router that is being configured, although a centrally managed solution is often required.

There is not a standard mechanism by which certificates are defined as authorized for some tasks and not for others. This authorization information can be captured in the certificate itself if the application is aware of the certificate-based authorization information. But this solution does not provide a simple mechanism for real-time updates to the authorization information and forces each application to be aware of the specific authorization information embedded in the certificate.

When the certificate-based access control list (ACL) mechanism is configured as part of the trustpoint authentication, the application is no longer responsible for determining this authorization information, and it is no longer possible to specify for which application the certificate is authorized. In some cases, the certificate-based ACL on the router gets so large that it cannot be managed. Additionally, it is beneficial to retrieve certificate-based ACL indications from an external server.

Current solutions to the real-time authorization problem involve specifying a new protocol and building a new server (with associated tasks, such as management and data distribution).

## PKI and AAA Server Integration for Certificate Status

Integrating your PKI with an authentication, authorization, and accounting (AAA) server provides an alternative online certificate status solution that leverages the existing AAA infrastructure. Certificates can be listed in the AAA database with appropriate levels of authorization. For components that do not explicitly support PKI-AAA, a default label of “all” from the AAA server provides authorization. Likewise, a label of “none” from the AAA database indicates that the specified certificate is not valid. (The absence of any application label is equivalent, but “none” is included for completeness and clarity). If the application component does support PKI-AAA, the component may be specified directly; for example, the application component could be “ipsec,” “ssl,” or “osp.” (ipsec=IP Security, ssl=Secure Sockets Layer, and osp=Open Settlement Protocol.)

**Note** Currently, no application component supports specification of the application label.

- There may be a time delay when accessing the AAA server. If the AAA server is not available, the authorization fails.

## RADIUS or TACACS+ Choosing a AAA Server Protocol

The AAA server can be configured to work with either the RADIUS or TACACS+ protocol. When you are configuring the AAA server for the PKI integration, you must set the RADIUS or TACACS attributes that are required for authorization.

If the RADIUS protocol is used, the password that is configured for the username in the AAA server should be set to “cisco,” which is acceptable because the certificate validation provides authentication and the AAA database is only being used for authorization. When the TACACS protocol is used, the password that is configured for the username in the AAA server is irrelevant because TACACS supports authorization without requiring authentication (the password is used for authentication).

In addition, if you are using TACACS, you must add a PKI service to the AAA server. The custom attribute “cert-application=all” is added under the PKI service for the particular user or usergroup to authorize the specific username.

## Attribute-Value Pairs for PKI and AAA Server Integration

The table below lists the attribute-value (AV) pairs that are to be used when setting up PKI integration with a AAA server. (Note the values shown in the table are possible values.) The AV pairs must match the client configuration. If they do not match, the peer certificate is not authorized.

**Note** Users can sometimes have AV pairs that are different from those of every other user. As a result, a unique username is required for each user. The **all** parameter (within the **authorization username** command) specifies that the entire subject name of the certificate will be used as the authorization username.

***Table 54: AV Pairs That Must Match***

| AV Pair | Value |
|---|---|
| cisco-avpair=pki:cert-application=all | Valid values are “all” and “none.” |
| cisco-avpair=pki:cert-trustpoint=msca | The value is a Cisco IOS XE command-line interface (CLI) configuration trustpoint label.<br>**Note**<br>The cert-trustpoint AV pair is normally optional. If it is specified, the device query must be coming from a certificate trustpoint that has a matching label, and the certificate that is authenticated must have the specified certificate serial number. |
| cisco-avpair=pki:cert-serial=16318DB7000100001671 | The value is a certificate serial number.<br>**Note**<br>The cert-serial AV pair is normally optional. If it is specified, the Cisco device query must be coming from a certificate trustpoint that has a matching label, and the certificate that is authenticated must have the specified certificate serial number. |

| AV Pair | Value |
|---|---|
| cisco-avpair=pki:cert-lifetime-end=1:00 jan 1, 2003 | The cert-lifetime-end AV pair is available to artificially extend a certificate lifetime beyond the time period that is indicated in the certificate itself. If the cert-lifetime-end AV pair is used, the cert-trustpoint and cert-serial AV pairs must also be specified. The value must match the following form: hours:minutes month day, year.<br>**Note**<br>Only the first three characters of a month are used: Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec. If more than three characters are entered for the month, the remaining characters are ignored (for example Janxxxx). |

## CRLs or OCSP Server Choosing a Certificate Revocation Mechanism

After a certificate is validated as a properly signed certificate, a certificate revocation method is performed to ensure that the certificate has not been revoked by the issuing CA. Cisco IOS XE software supports two revocation mechanisms--certificate revocation lists (CRLs) and Online Certificate Status Protocol (OCSP). Cisco IOS XE software also supports AAA integration for certificate checking; however, additional authorization functionality is included. For more information on PKI and AAA certificate authorization and status check, see the PKI and AAA Server Integration for Certificate Status section.

The following sections explain how each revocation mechanism works:

### What Is a CRL

A certificate revocation list (CRL) is a list of revoked certificates. The CRL is created and digitally signed by the CA that originally issued the certificates. The CRL contains dates for when each certificate was issued and when it expires.

CAs publish new CRLs periodically or when a certificate for which the CA is responsible has been revoked. By default, a new CRL is downloaded after the currently cached CRL expires. An administrator may also configure the duration for which CRLs are cached in router memory or disable CRL caching completely. The CRL caching configuration applies to all CRLs associated with a trustpoint.

When the CRL expires, the router deletes it from its cache. A new CRL is downloaded when a certificate is presented for verification; however, if a newer version of the CRL that lists the certificate under examination is on the server but the router is still using the CRL in its cache, the router does not know that the certificate has been revoked. The certificate passes the revocation check even though it should have been denied.

When a CA issues a certificate, the CA can include in the certificate the CRL distribution point (CDP) for that certificate. Cisco IOS client devices use CDPs to locate and load the correct CRL. The Cisco IOS client supports multiple CDPs, but the Cisco IOS CA currently supports only one CDP; however, third-party vendor CAs may support multiple CDPs or different CDPs per certificate. If a CDP is not specified in the certificate, the client device uses the default Simple Certificate Enrollment Protocol (SCEP) method to retrieve the CRL. (The CDP location can be specified through the **cdp-url**command.)

When implementing CRLs, you should consider the following design considerations:

- CRL lifetimes and the security association (SA) and Internet Key Exchange (IKE) lifetimes.

- The CRL lifetime determines the length of time between CA-issued updates to the CRL. The default CRL lifetime value, which is 168 hours [1 week], can be changed through the **lifetime crl** command.
- The method of the CDP determines how the CRL is retrieved; some possible choices include HTTP, Lightweight Directory Access Protocol (LDAP), SCEP, or TFTP. HTTP, TFTP, and LDAP are the most commonly used methods. Although Cisco IOS software defaults to SCEP, an HTTP CDP is recommended for large installations using CRLs because HTTP can be made highly scalable.
- The location of the CDP determines from where the CRL is retrieved; for example, you can specify the server and file path from which to retrieve the CRL.

### *Querying All CDPs During Revocation Check*

When a CDP server does not respond to a request, the Cisco IOS XE software reports an error, which may result in the peer's certificate being rejected. To prevent a possible certificate rejection and if there are multiple CDPs in a certificate, the Cisco IOS XE software will attempt to use the CDPs in the order in which they appear in the certificate. The device will attempt to retrieve a CRL using each CDP URL or directory specification. If an error occurs using a CDP, an attempt will be made using the next CDP.

**Tip** Although the Cisco IOS XE software will make every attempt to obtain the CRL from one of the indicated CDPs, it is recommended that you use an HTTP CDP server with high-speed redundant HTTP servers to avoid application timeouts because of slow CDP responses.

## What Is OCSP

OCSP is an online mechanism that is used to determine certificate validity and provides the following flexibility as a revocation mechanism:

- OCSP can provide real-time certificate status checking.
- OCSP allows the network administrator to specify a central OCSP server, which can service all devices within a network.
- OCSP also allows the network administrator the flexibility to specify multiple OCSP servers, either per client certificate or per group of client certificates.
- OCSP server validation is usually based on the root CA certificate or a valid subordinate CA certificate, but may also be configured so that external CA certificates or self-signed certificates may be used. Using external CA certificates or self-signed certificates allows the OCSP servers certificate to be issued and validated from an alternative PKI hierarchy.

A network administrator can configure an OCSP server to collect and update CRLs from different CA servers. The devices within the network can rely on the OCSP server to check the certificate status without retrieving and caching each CRL for every peer. When peers have to check the revocation status of a certificate, they send a query to the OCSP server that includes the serial number of the certificate in question and an optional unique identifier for the OCSP request, or a nonce. The OCSP server holds a copy of the CRL to determine if the CA has listed the certificate as being revoked; the server then responds to the peer including the nonce. If the nonce in the response from the OCSP server does not match the original nonce sent by the peer, the response is considered invalid and certificate verification fails. The dialog between the OCSP server and the peer consumes less bandwidth than most CRL downloads.

If the OCSP server is using a CRL, CRL time limitations will be applicable; that is, a CRL that is still valid might be used by the OCSP server although a new CRL has been issued by the CRL containing additional

certificate revocation information. Because fewer devices are downloading the CRL information on a regular basis, you can decrease the CRL lifetime value or configure the OCSP server not to cache the CRL. For more information, check your OCSP server documentation.

**Note** OCSP multiple response handling: Support has been enabled for handling of multiple OCSP single

responses from an OCSP responder in a response packet. In addition to the debug log messages the following debug log message will be displayed:

CRYPTO_PKI: Number of single Responses in OCSP response:1(this value can change depending upon the number of responses).

*When to Use an OCSP Server*

OCSP may be more appropriate than CRLs if your PKI has any of the following characteristics:

- Real-time certificate revocation status is necessary. CRLs are updated only periodically and the latest CRL may not always be cached by the client device. For example, if a client does not yet have the latest CRL cached and a newly revoked certificate is being checked, that revoked certificate will successfully pass the revocation check.
- There are a large number of revoked certificates or multiple CRLs. Caching a large CRL consumes large portions of Cisco IOS memory and may reduce resources available to other processes.
- CRLs expire frequently, causing the CDP to handle a larger load of CRLs.

## When to Use Certificate-Based ACLs for Authorization or Revocation

Certificates contain several fields that are used to determine whether a device or user is authorized to perform a specified action.

Because certificate-based ACLs are configured on the device, they do not scale well for large numbers of ACLs; however, certificate-based ACLs do provide very granular control of specific device behavior. Certificate-based ACLs are also leveraged by additional features to help determine when PKI components such as revocation, authorization, or a trustpoint should be used. They provide a general mechanism allowing users to select a specific certificate or a group of certificates that are being validated for either authorization or additional processing.

Certificate-based ACLs specify one or more fields within the certificate and an acceptable value for each specified field. You can specify which fields within a certificate should be checked and which values those fields may or may not have.

There are six logical tests for comparing the field with the value--equal, not equal, contains, does not contain, less than, and greater than or equal. If more than one field is specified within a single certificate-based ACL, the tests of all of the fields within the ACL must succeed to match the ACL. The same field may be specified multiple times within the same ACL. More than one ACL may be specified, and ACL will be processed in turn until a match is found or all of the ACLs have been processed.

### Ignore Revocation Checks Using a Certificate-Based ACL

Certificate-based ACLs can be configured to instruct your router to ignore the revocation check and expired certificates of a valid peer. Thus, a certificate that meets the specified criteria can be accepted regardless of the validity period of the certificate, or if the certificate meets the specified criteria, revocation checking does

not have to be performed. You can also use a certificate-based ACL to ignore the revocation check when the communication with a AAA server is protected with a certificate.

### Ignoring Revocation Lists

To allow a trustpoint to enforce CRLs except for specific certificates, enter the **match certificate**command with the **skip revocation-check** keyword. This type of enforcement is most useful in a hub-and-spoke configuration in which you also want to allow direct spoke-to-spoke connections. In pure hub-and-spoke configurations, all spokes connect only to the hub, so CRL checking is necessary only on the hub. For one spoke to communicate directly with another spoke, the **match certificate**command with the **skip revocation-check** keyword can be used for neighboring peer certificates instead of requiring a CRL on each spoke.

### Ignoring Expired Certificates

To configure your router to ignore expired certificates, enter the **match certificate** command with the **allow expired-certificate** keyword. This command has the following purposes:

- If the certificate of a peer has expired, this command may be used to “allow” the expired certificate until the peer can obtain a new certificate.
- If your router clock has not yet been set to the correct time, the certificate of a peer will appear to be not yet valid until the clock is set. This command may be used to allow the certificate of the peer even though your router clock is not set.

**Note** If Network Time Protocol (NTP) is available only via the IPSec connection (usually via the hub in a hub-and-spoke configuration), the router clock can never be set. The tunnel to the hub cannot be “brought up” because the certificate of the hub is not yet valid.

- “Expired” is a generic term for a certificate that is expired or that is not yet valid. The certificate has a start and end time. An expired certificate, for purposes of the ACL, is one for which the current time of the router is outside the start and end times specified in the certificate.

### Skipping the AAA Check of the Certificate

If the communication with an AAA server is protected with a certificate, and you want to skip the AAA check of the certificate, use the **match certificate** command with the **skip authorization-check** keyword. For example, if a virtual private network (VPN) tunnel is configured so that all AAA traffic goes over that tunnel, and the tunnel is protected with a certificate, you can use the **match certificate** command with the **skip authorization-check** keyword to skip the certificate check so that the tunnel can be established.

The **match certificate**command and the **skip authorization-check** keyword should be configured after PKI integration with an AAA server is configured.

**Note** If the AAA server is available only via an IPSec connection, the AAA server cannot be contacted until after the IPSec connection is established. The IPSec connection cannot be “brought up” because the certificate of the AAA server is not yet valid.

## PKI Certificate Chain Validation

A certificate chain establishes a sequence of trusted certificates --from a peer certificate to the root CA certificate. Within a PKI hierarchy, all enrolled peers can validate the certificate of one another if the peers share a trusted root CA certificate or a common subordinate CA. Each CA corresponds to a trustpoint.

When a certificate chain is received from a peer, the default processing of a certificate chain path continues until the first trusted certificate, or trustpoint, is reached. An administrator may configure the level to which a certificate chain is processed on all certificates including subordinate CA certificates.

Configuring the level to which a certificate chain is processed allows for the reauthentication of trusted certificates, the extension of a trusted certificate chain, and the completion of a certificate chain that contains a gap.

#### Reauthentication of Trusted Certificates

The default behavior is for the device to remove any trusted certificates from the certificate chain sent by the peer before the chain is validated. An administrator may configure certificate chain path processing so that the device does not remove CA certificates that are already trusted before chain validation, so that all certificates in the chain are re-authenticated for the current session.

#### Extending the Trusted Certificate Chain

The default behavior is for the device to use its trusted certificates to extend the certificate chain if there are any missing certificates in the certificate chain sent by the peer. The device will validate only certificates in the chain sent by the peer. An administrator may configure certificate chain path processing so that the certificates in the peer's certificate chain and the device's trusted certificates are validated to a specified point.

#### Completing Gaps in a Certificate Chain

An administrator may configure certificate chain processing so that if there is a gap in the configured trustpoint hierarchy, certificates sent by the peer can be used to complete the set of certificates to be validated.

**Note** If the trustpoint is configured to require parent validation and the peer does not provide the full certificate chain, the gap cannot be completed and the certificate chain is rejected and invalid.

**Note** It is a configuration error if the trustpoint is configured to require parent validation and there is no parent trustpoint configured. The resulting certificate chain gap cannot be completed and the subordinate CA certificate cannot be validated. The certificate chain is invalid.

# How to Configure Authorization and Revocation of Certificates in a PKI

## Configuring PKI Integration with a AAA Server

Perform this task to generate a AAA username from the certificate presented by the peer and specify which fields within a certificate should be used to build the AAA database username.

**Note** The following restrictions should be considered when using the **all** keyword as the subject name for the **authorization username** command:

- Some AAA servers limit the length of the username (for example, to 64 characters). As a result, the entire certificate subject name cannot be longer than the limitation of the server.
- Some AAA servers limit the available character set that may be used for the username (for example, a space [ ] and an equal sign [=] may not be acceptable). You cannot use the **all** keyword for a AAA server having such a character-set limitation.
- The **subject-name** command in the trustpoint configuration may not always be the final AAA subject name. If the fully qualified domain name (FQDN), serial number, or IP address of the router are included in a certificate request, the subject name field of the issued certificate will also have these components. To turn off the components, use the **fqdn**, **serial-number**, and **ip-address** commands with the **none** keyword.
- CA servers sometimes change the requested subject name field when they issue a certificate. For example, CA servers of some vendors switch the relative distinguished names (RDNs) in the requested subject names to the following order: CN, OU, O, L, ST, and C. However, another CA server might append the configured LDAP directory root (for example, O=cisco.com) to the end of the requested subject name.
- Depending on the tools you choose for displaying a certificate, the printed order of the RDNs in the subject name could be different. Cisco IOS software always displays the least significant RDN first, but other software, such as Open Source Secure Socket Layer (OpenSSL), does the opposite. Therefore, if you are configuring a AAA server with a full distinguished name (DN) (subject name) as the corresponding username, ensure that the Cisco IOS software style (that is, with the least significant RDN first) is used.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password, if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables the AAA access control model. |
| **Step 4** | **aaa authorization network** *listname* [*method*]<br>**Example:**<br>`Device(config)# aaa authorization network maxaaa group tacacs+` | Sets the parameters that restrict user access to a network.<br>• *method* : Can be **group radius**, **group tacacs+**, or **group group-name**. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **crypto pki trustpoint** *name*<br>**Example:**<br>`Device(config)# crypto pki trustpoint msca` | Declares the trustpoint and a given name and enters ca-trustpoint configuration mode. |
| **Step 6** | **enrollment** [**mode**] [**retry period** *minutes*] [**retry count** *number*] **url** *url* [**pem**]<br>**Example:**<br>`Device(ca-trustpoint)# enrollment url http://caserver.myexample.com`<br>- or-<br>`Device(ca-trustpoint)# enrollment url http://[2001:DB8:1:1::1]:80` | Specifies the following enrollment parameters of the CA:<br>• (Optional) The **mode** keyword specifies the registration authority (RA) mode, if your CA system provides an RA. By default, RA mode is disabled.<br>• (Optional) The **retry period** keyword and *minutes* argument specifies the period, in minutes, in which the router waits before sending the CA another certificate request. Valid values are from 1 to 60. The default is 1.<br>• (Optional) The **retry count** keyword and *number* argument specifies the number of times a router will resend a certificate request when it does not receive a response from the previous request. Valid values are from 1 to 100. The default is 10.<br>• The *url* argument is the URL of the CA to which your router should send certificate requests.<br>**Note**<br>An IPv6 address can be added to the **http:** enrollment method. For example: http://[ipv6-address]:80. The IPv6 address must be enclosed in brackets in the URL.<br>• (Optional) The **pem** keyword adds privacy-enhanced mail (PEM) boundaries to the certificate request. |
| **Step 7** | revocation-check method<br>**Example:**<br>`Device(ca-trustpoint)# revocation-check crl` | (Optional) Checks the revocation status of a certificate. |
| **Step 8** | **exit**<br>**Example:** | Exits ca-trustpoint configuration mode and returns to global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(ca-trustpoint)# exit` | |
| **Step 9** | **authorization username subjectname** *subjectname*<br>**Example:**<br>`Device(config)# authorization username subjectname serialnumber` | Sets parameters for the different certificate fields that are used to build the AAA username.<br>The *subjectname* argument can be any of the following:<br>• **all**: Entire distinguished name (subject name) of the certificate.<br>• **commonname**: Certification common name.<br>• **country**: Certificate country.<br>• **email**: Certificate e-mail.<br>• **ipaddress**: Certificate IP address.<br>• **locality**: Certificate locality.<br>• **organization**: Certificate organization.<br>• **organizationalunit**: Certificate organizational unit.<br>• **postalcode**: Certificate postal code.<br>• **serialnumber**: Certificate serial number.<br>• **state**: Certificate state field.<br>• **streetaddress**: Certificate street address.<br>• **title**: Certificate title.<br>• **unstructuredname**: Certificate unstructured name. |
| **Step 10** | **authorization list** *listname*<br>**Example:**<br>`Device(config)# authorization list maxaaa` | Specifies the AAA authorization list. |
| **Step 11** | **tacacs server** *server-name*<br>**Example:**<br>`Device(config)# tacacs server yourserver` | Specifies a TACACS+ server. |
| **Step 12** | **address** {**ipv4** \| **ipv6**} *ip-address*<br>**Example:**<br>`Device(config-server-tacacs)# address ipv4 192.0.2.2` | Configures the IP address for the TACACS server. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 13** | **key** *string*<br>**Example:**<br>`Device(config-server-tacacs)# key a_secret_key` | Configures the authorization and encryption key used between the switch and the TACACS server. |
| **Step 14** | **end**<br>**Example:**<br>`Device(config-server-tacacs)# end`<br>**Example:** | Returns to privileged EXEC mode. |

## Troubleshooting Tips

To display debug messages for the trace of interaction (message type) between the CA and the router, use the **debug crypto pki transactions**command. (See the sample output, which shows a successful PKI integration with AAA server exchange and a failed PKI integration with AAA server exchange.)

### Successful Exchange

```
Device# debug crypto pki transactions

Apr 22 23:15:03.695: CRYPTO_PKI: Found a issuer match
Apr 22 23:15:03.955: CRYPTO_PKI: cert revocation status unknown.
Apr 22 23:15:03.955: CRYPTO_PKI: Certificate validated without revocation check
```

Each line that shows “CRYPTO_PKI_AAA” indicates the state of the AAA authorization checks. Each of the AAA AV pairs is indicated, and then the results of the authorization check are shown.

```
Apr 22 23:15:04.019: CRYPTO_PKI_AAA: checking AAA authorization (ipsecca_script_aaalist,
PKIAAA-L, <all>)
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: reply attribute ("cert-application" = "all")
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: reply attribute ("cert-trustpoint" = "CA1")
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: reply attribute ("cert-serial" = "15DE")
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: authorization passed
Apr 22 23:12:30.327: CRYPTO_PKI: Found a issuer match
```

### Failed Exchange

```
Device# debug crypto pki transactions

Apr 22 23:11:13.703: CRYPTO_PKI_AAA: checking AAA authorization =
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: reply attribute ("cert-application" = “all”)
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: reply attribute ("cert-trustpoint"= “CA1”)
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: reply attribute ("cert-serial" = “233D”)
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: parsed cert-lifetime-end as: 21:30:00
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: timezone specific extended
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: cert-lifetime-end is expired
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: cert-lifetime-end check failed.
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: authorization failed
```

In the above failed exchange, the certificate has expired.

# Configuring a Revocation Mechanism for PKI Certificate Status Checking

Perform this task to set up a CRL as the certificate revocation mechanism--CRLs or OCSP--that is used to check the status of certificates in a PKI.

### The revocation-check Command

Use the **revocation-check** command to specify at least one method (OCSP, CRL, or skip the revocation check) that is to be used to ensure that the certificate of a peer has not been revoked. For multiple methods, the order in which the methods are applied is determined by the order specified via this command.

If your device does not have the applicable CRL and is unable to obtain one or if the OCSP server returns an error, your device will reject the peer's certificate--unless you include the **none** keyword in your configuration. If the **none** keyword is configured, a revocation check will not be performed and the certificate will always be accepted.

### Nonces and Peer Communications with OCSP Servers

When using OCSP, nonces, unique identifiers for OCSP requests, are sent by default during peer communications with your OCSP server. The use of nonces offers a more secure and reliable communication channel between the peer and OCSP server.

If your OCSP server does not support nonces, you may disable the sending of nonces. For more information, see your OCSP server documentation.

#### Before you begin

- Before issuing any client certificates, the appropriate settings on the server (such as setting the CDP) should be configured.
- When configuring an OCSP server to return the revocation status for a CA server, the OCSP server must be configured with an OCSP response signing certificate that is issued by that CA server. Ensure that the signing certificate is in the correct format, or the router will not accept the OCSP response. See your OCSP manual for additional information.

**Note**

- OCSP transports messages over HTTP, so there may be a time delay when you access the OCSP server.
- If the OCSP server depends on normal CRL processing to check revocation status, the same time delay that affects CRLs will also apply to OCSP.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **crypto pki trustpoint** *name*<br>**Example:**<br>`Device(config)# crypto pki trustpoint hazel` | Declares the trustpoint and a given name and enters ca-trustpoint configuration mode. |
| **Step 4** | **ocsp url** *url*<br>**Example:**<br>`Device(ca-trustpoint)# ocsp url http://ocsp-server`<br>- or -<br>`Device(ca-trustpoint)# ocsp url http://10.10.10.1:80`<br>- or -<br>`Device(ca-trustpoint)# ocsp url http://[2001DB8:1:1::2]:80` | The *url* argument specifies the URL of an OCSP server so that the trustpoint can check the certificate status. This URL overrides the URL of the OCSP server (if one exists) in the Authority Info Access (AIA) extension of the certificate. All certificates associated with a configured trustpoint are checked by the OCSP server. The URL can be a hostname, IPv4 address, or an IPv6 address. |
| **Step 5** | **revocation-check** *method1* [*method2 method3*]]<br>**Example:**<br>`Device(ca-trustpoint)# revocation-check ocsp none` | Checks the revocation status of a certificate.<br>• **crl** —Certificate checking is performed by a CRL. This is the default option.<br>• **none** —Certificate checking is ignored.<br>• **ocsp** —Certificate checking is performed by an OCSP server.<br>If a second and third method are specified, each method will be used only if the previous method returns an error, such as a server being down. |
| **Step 6** | **ocsp disable-nonce**<br>**Example:**<br>`Device(ca-trustpoint)# ocsp disable-nonce` | (Optional) Specifies that a nonce, or an OCSP request unique identifier, will not be sent during peer communications with the OCSP server. |
| **Step 7** | **end**<br>**Example:**<br>`Device(ca-trustpoint)# end` | Exits ca-trustpoint configuration mode and returns to privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **show crypto pki certificates**<br>**Example:**<br>`Device# show crypto pki certificates` | (Optional) Displays information about your certificates. |
| **Step 9** | **show crypto pki trustpoints** [**status** \| *label* [**status**]]<br>**Example:**<br>`Device# show crypto pki trustpoints` | Displays information about the trustpoint configured in router. |

## Configuring Certificate Authorization and Revocation Settings

Perform this task to specify a certificate-based ACL, to ignore revocation checks or expired certificates, to manually override the default CDP location, to manually override the OCSP server setting, to configure CRL caching, or to set session acceptance or rejection based on a certificate serial number, as appropriate.

### Configuring Certificate-Based ACLs to Ignore Revocation Checks

To configure your router to use certificate-based ACLs to ignore revocation checks and expired certificates, perform the following steps:

- Identify an existing trustpoint or create a new trustpoint to be used when verifying the certificate of the peer. Authenticate the trustpoint if it has not already been authenticated. The router may enroll with this trustpoint if you want. Do not set optional CRLs for the trustpoint if you plan to use the **match certificate** command and **skip revocation-check** keyword.
- Determine the unique characteristics of the certificates that should not have their CRL checked and of the expired certificates that should be allowed.
- Define a certificate map to match the characteristics identified in the prior step.
- You can add the **match certificate** command and **skip revocation-check** keyword and the **match certificate command** and **allow expired-certificate** keyword to the trustpoint that was created or identified in the first step.

**Note** Certificate maps are checked even if the peer's public key is cached. For example, when the public key is cached by the peer, and a certificate map is added to the trustpoint to ban a certificate, the certificate map is effective. This prevents a client with the banned certificate, which was once connected in the past, from reconnecting.

### Manually Overriding CDPs in a Certificate

Users can override the CDPs in a certificate with a manually configured CDP. Manually overriding the CDPs in a certificate can be advantageous when a particular server is unavailable for an extended period of time. The certificate's CDPs can be replaced with a URL or directory specification without reissuing all of the certificates that contain the original CDP.

## Manually Overriding the OCSP Server Setting in a Certificate

Administrators can override the OCSP server setting specified in the Authority Information Access ( AIA) field of the client certificate or set by the issuing the **ocsp url** command. One or more OCSP servers may be manually specified, either per client certificate or per group of client certificates by the **match certificate override ocsp** command. The **match certificate override ocsp**command overrides the client certificate AIA field or the **ocsp url**command setting if a client certificate is successfully matched to a certificate map during the revocation check.

**Note** Only one OCSP server can be specified per client certificate.

## Configuring CRL Cache Control

By default, a new CRL will be downloaded after the currently cached CRL expires. Administrators can either configure the maximum amount of time in minutes a CRL remains in the cache by issuing the **crl cache delete-after** command or disable CRL caching by issuing the **crl cache none** command. Only the **crl-cache delete-after**command or the **crl-cache none** command may be specified. If both commands are entered for a trustpoint, the last command executed will take effect and a message will be displayed.

Neither the **crl-cache none** command nor the **crl-cache delete-after** command affects the currently cached CRL. If you configure the **crl-cache none** command, all CRLs downloaded after this command is issued will not be cached. If you configure the **crl-cache delete-after** command, the configured lifetime will only affect CRLs downloaded after this command is issued.

This functionality is useful is when a CA issues CRLs with no expiration date or with expiration dates days or weeks ahead.

## Configuring Certificate Serial Number Session Control

A certificate serial number can be specified to allow a certificate validation request to be accepted or rejected by the trustpoint for a session. A session may be rejected, depending on certificate serial number session control, even if a certificate is still valid. Certificate serial number session control may be configured by using either a certificate map with the **serial-number** field or an AAA attribute, with the **cert-serial-not** command.

Using certificate maps for session control allows an administrator to specify a single certificate serial number. Using the AAA attribute allows an administrator to specify one or more certificate serial numbers for session control.

### Before you begin

- The trustpoint should be defined and authenticated before attaching certificate maps to the trustpoint.
- The certificate map must be configured before the CDP override feature can be enabled or the **serial-number** command is issued.
- The PKI and AAA server integration must be successfully completed to use AAA attributes as described in “PKI and AAA Server Integration for Certificate Status.”

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **crypto pki certificate map** *label sequence-number*<br>**Example:**<br>`Device(config)# crypto pki certificate map Group 10` | Defines values in a certificate that should be matched or not matched and enters ca-certificate-map configuration mode. |
| **Step 4** | *field-name match-criteria match-value*<br>**Example:**<br>`Device(ca-certificate-map)# subject-name co MyExample` | Specifies one or more certificate fields together with their matching criteria and the value to match.<br>The *field-name* is one of the following case-insensitive name strings or a date:<br>• **alt-subject-name**<br>• **expires-on**<br>• **issuer-name**<br>• **name**<br>• **serial-number**<br>• **subject-name**<br>• **unstructured-subject-name**<br>• **valid-start**<br>**Note**<br>Date field format is dd mm yyyy hh:mm:ss or mmm dd yyyy hh:mm:ss.<br>The *match-criteria* is one of the following logical operators:<br>• **co** —contains (valid only for name fields and serial number field) |

| | Command or Action | Purpose |
|---|---|---|
| | | • **eq** —equal (valid for name, serial number, and date fields)<br>• **ge** —greater than or equal (valid only for date fields)<br>• **lt** —less than (valid only for date fields)<br>• **nc** —does not contain (valid only for name fields and serial number field)<br>• **ne** —not equal (valid for name, serial number, and date fields)<br><br>The *match-value*is the name or date to test with the logical operator assigned by match-criteria.<br><br>**Note**<br>Use this command only when setting up a certificate-based ACL—not when setting up a certificate-based ACL to ignore revocation checks or expired certificates. |
| **Step 5** | **exit**<br>**Example:**<br>`Device(ca-certificate-map)# exit` | Exits ca-certificate-map configuration mode and returns to global configuration mode. |
| **Step 6** | **crypto pki trustpoint** *name*<br>**Example:**<br>`Device(config)# crypto pki trustpoint Access2` | Declares the trustpoint, given name and enters ca-trustpoint configuration mode. |
| **Step 7** | Do one of the following:<br>• **crl-cache none**<br>• **crl-cache delete-after** *time*<br>**Example:**<br>`Device(ca-trustpoint)# crl-cache none`<br>**Example:**<br>`Device(ca-trustpoint)# crl-cache delete-after 20` | (Optional) Disables CRL caching completely for all CRLs associated with the trustpoint.<br><br>The **crl-cache none** command does not affect any currently cached CRLs. All CRLs downloaded after this command is configured will not be cached.<br><br>(Optional) Specifies the maximum time CRLs will remain in the cache for all CRLs associated with the trustpoint.<br>• *time* —The amount of time in minutes before the CRL is deleted.<br><br>The **crl-cache delete-after** command does not affect any currently cached CRLs. The configured lifetime will only affect CRLs downloaded after this command is configured. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **match certificate** *certificate-map-label* [**allow expired-certificate** \| **skip revocation-check** \| **skip authorization-check**<br>**Example:**<br>`Device(ca-trustpoint)# match certificate Group skip revocation-check` | (Optional) Associates the certificate-based ACL (that was defined via the **crypto pki certificate map** command) to a trustpoint.<br>• *certificate-map-label* —Must match the *label* argument specified via the **crypto pki certificate map** command.<br>• **allowexpired-certificate** —Ignores expired certificates.<br>• **skip revocation-check** —Allows a trustpoint to enforce CRLs except for specific certificates.<br>• **skip authorization-check** —Skips the AAA check of a certificate when PKI integration with an AAA server is configured. |
| **Step 9** | **match certificate** *certificate-map-label* **override cdp** {**url** \| **directory**} *string*<br>**Example:**<br>`Device(ca-trustpoint)# match certificate Group1 override cdp url http://server.cisco.com` | (Optional) Manually overrides the existing CDP entries for a certificate with a URL or directory specification.<br>• *certificate-map-label* —A user-specified label that must match the *label* argument specified in a previously defined **crypto pki certificate map** command.<br>• **url** —Specifies that the certificate's CDPs will be overridden with an HTTP or LDAP URL.<br>• **directory** —Specifies that the certificate's CDPs will be overridden with an LDAP directory specification.<br>• *string* —The URL or directory specification.<br>**Note**<br>Some applications may time out before all CDPs have been tried and will report an error message. The error message will not affect the router, and the Cisco IOS software will continue attempting to retrieve a CRL until all CDPs have been tried. |
| **Step 10** | **match certificate** *certificate-map-label* **override ocsp** [**trustpoint** *trustpoint-label*] *sequence-number* **url** *ocsp-url*<br>**Example:** | (Optional) Specifies an OCSP server, either per client certificate or per group of client certificates, and may be issued more than once to specify additional OCSP servers and client |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(ca-trustpoint)# match certificate mycertmapname override ocsp trustpoint mytp 15 url http://192.0.2.2` | certificate settings including alternative PKI hierarchies.<br>• *certificate-map-label* —The name of an existing certificate map.<br>• **trustpoint** —The trustpoint to be used when validating the OCSP server certificate.<br>• *sequence-number* —The order the **match certificate override ocsp** command statements apply to the certificate being verified. Matches are performed from the lowest sequence number to the highest sequence number. If more than one command is issued with the same sequence number, it overwrites the previous OCSP server override setting.<br>• **url** —The URL of the OCSP server.<br>When the certificate matches a configured certificate map, the AIA field of the client certificate and any previously issued **ocsp url** command settings are overwritten with the specified OCSP server.<br>If no map-based match occurs, one of the following two cases will continue to apply to the client certificate.<br>• If OCSP is specified as the revocation method, the AIA field value will continue to apply to the client certificate.<br>• If the **ocsp url** configuration exists, the **ocsp url** configuration settings will continue to apply to the client certificates. |
| **Step 11** | **exit**<br>**Example:**<br>`Device(ca-trustpoint)# exit` | Exits Returns to global configuration mode. |
| **Step 12** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | (Optional) Enables the AAA access control model. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 13** | **aaa attribute list** *list-name*<br>**Example:**<br>`Device(config)# aaa attribute list crl` | (Optional) Defines an AAA attribute list locally on a router and enters config-attr-list configuration mode. |
| **Step 14** | **attribute type** {*name*} {*value*}<br>**Example:**<br>`Device(config-attr-list)# attribute type cert-serial-not 6C4A` | (Optional) Defines an AAA attribute type that is to be added to an AAA attribute list locally on a router.<br>To configure certificate serial number session control, an administrator may specify a specific certificate in the *value* field to be accepted or rejected based on its serial number where *name* is set to **cert-serial-not**. If the serial number of the certificate matches the serial number specified by the attribute type setting, the certificate will be rejected.<br>For a full list of available AAA attribute types, execute the **show aaa attributes** command. |
| **Step 15** | **exit**<br>**Example:**<br>`Device(ca-trustpoint)# end`<br>**Example:**<br>`Device(config-attr-list)# end` | Returns to privileged EXEC mode. |
| **Step 16** | **show crypto pki certificates**<br>**Example:**<br>`Device# show crypto pki certificates` | (Optional) Displays the components of the certificates installed on the router if the CA certificate has been authenticated. |

### Example

The following is a sample certificate. The OCSP-related extensions are shown using exclamation points.

```
Certificate:
        Data:
            Version: v3
            Serial Number:0x14
            Signature Algorithm:SHAwithRSA - 1.2.840.113549.1.1.4
            Issuer:CN=CA server,OU=PKI,O=Cisco Systems
            Validity:
                Not Before:Thursday, August 8, 2002 4:38:05 PM PST
                Not After:Tuesday, August 7, 2003 4:38:05 PM PST
            Subject:CN=OCSP server,OU=PKI,O=Cisco Systems
            Subject Public Key Info:
```

```
                Algorithm:RSA - 1.2.840.113549.1.1.1
                Public Key:
                    Exponent:65537
                    Public Key Modulus:(2048 bits) :
                        <snip>
            Extensions:
                Identifier:Subject Key Identifier - 2.5.29.14
                    Critical:no
                    Key Identifier:
                        <snip>
                Identifier:Authority Key Identifier - 2.5.29.35
                    Critical:no
                    Key Identifier:
                        <snip>
!                Identifier:OCSP NoCheck:- 1.3.6.1.5.5.7.48.1.5
                     Critical:no
                Identifier:Extended Key Usage:- 2.5.29.37
                     Critical:no
                     Extended Key Usage:
                     OCSPSigning
!
                Identifier:CRL Distribution Points - 2.5.29.31
                    Critical:no
                    Number of Points:1
                    Point 0
                        Distribution Point:
[URIName:ldap://CA-server/CN=CA server,OU=PKI,O=Cisco Systems]
        Signature:
            Algorithm:SHAwithRSA - 1.2.840.113549.1.1.4
            Signature:
            <snip>
```

The following example shows an excerpt of the running configuration output when adding a **match certificate override ocsp** command to the beginning of an existing sequence:

```
match certificate map3 override ocsp 5 url http://192.0.2.3/
show running-configuration
.
.
.
        match certificate map3 override ocsp 5 url http://192.0.2.3/
        match certificate map1 override ocsp 10 url http://192.0.2.1/
        match certificate map2 override ocsp 15 url http://192.0.2.2/
```

The following example shows an excerpt of the running configuration output when an existing **match certificate override ocsp** command is replaced and a trustpoint is specified to use an alternative PKI hierarchy:

```
match certificate map4 override ocsp trustpoint tp4 10 url http://192.0.2.4/newvalue
show running-configuration
.
.
.
        match certificate map3 override ocsp trustpoint tp3 5 url http://192.0.2.3/
        match certificate map1 override ocsp trustpoint tp1 10 url http://192.0.2.1/
        match certificate map4 override ocsp trustpoint tp4 10 url
http://192.0.2.4/newvalue
        match certificate map2 override ocsp trustpoint tp2 15 url http://192.0.2.2/
```

### Troubleshooting Tips

If you ignored revocation check or expired certificates, you should carefully check your configuration. Verify that the certificate map properly matches either the certificate or certificates that should be allowed or the AAA checks that should be skipped. In a controlled environment, try modifying the certificate map and determine what is not working as expected.

## Configuring Certificate Chain Validation

Perform this task to configure the processing level for the certificate chain path of your peer certificates.

#### Before you begin

- The device must be enrolled in your PKI hierarchy.
- The appropriate key pair must be associated with the certificate.

**Note**

- A trustpoint associated with the root CA cannot be configured to be validated to the next level.

The **chain-validation** command is configured with the **continue** keyword for the trustpoint associated with the root CA, an error message will be displayed and the chain validation will revert to the default **chain-validation**command setting.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **crypto pki trustpoint***name*<br>**Example:**<br>`Device(config)# crypto pki trustpoint ca-sub1` | Declares the trustpoint and a given name and enters ca-trustpoint configuration mode. |
| **Step 4** | **chain-validation** [{**stop** \| **continue**} [*parent-trustpoint*]]<br>**Example:**<br>`Device(ca-trustpoint)# chain-validation continue ca-sub1` | Configures the level to which a certificate chain is processed on all certificates including subordinate CA certificates.<br>• Use the **stop**keyword to specify that the certificate is already trusted. This is the default setting. |

| | Command or Action | Purpose |
|---|---|---|
| | | • Use the **continue** keyword to specify that the that the subordinate CA certificate associated with the trustpoint must be validated.<br>• The *parent-trustpoint* argument specifies the name of the parent trustpoint the certificate must be validated against. |
| **Step 5** | **exit**<br>**Example:**<br>`Device(ca-trustpoint)# exit` | Exits ca-trustpoint configuration mode and returns to global configuration mode |

# Configuration Examples for Authorization and Revocation of Certificates in a PKI

## Configuration and Verification Examples fo PKI AAA Authorization

This section provides configuration examples of PKI AAA authorizations:

### Example: Device Configuration

The following **show running-config**command output shows the working configuration of a device that is set up to authorize VPN connections using the PKI Integration with AAA Server feature:

```
Device#show running-config

Building configuration...
!
version 16.8
!
hostname catxxxx
!
aaa new-model
!
!
aaa authentication login default group tacacs+
aaa authentication login no_tacacs enable
aaa authentication ppp default group tacacs+
aaa authorization exec ACSLab group tacacs+
aaa authorization network ACSLab group tacacs+
aaa accounting exec ACSLab start-stop group tacacs+
aaa accounting network default start-stop group ACSLab
aaa session-id common
!
ip domain name example.com
!
crypto pki trustpoint EM-CERT-SERV
 enrollment url http://192.0.2.33:80
 serial-number
 crl optional
 rsakeypair STOREVPN 2048
```

```
 auto-enroll
 authorization list ACSLab
!
crypto pki certificate chain EM-CERT-SERV
 certificate 04
  30820214 3082017D A0030201 02020104 300D0609 2A864886 F70D0101 04050030
  17311530 13060355 0403130C 454D2D43 4552542D 53455256 301E170D 30343031
  31393232 30323535 5A170D30 35303131 38323230 3235355A 3030312E 300E0603
  55040513 07314437 45424434 301C0609 2A864886 F70D0109 02160F37 3230302D
  312E6772 696C2E63 6F6D3081 9F300D06 092A8648 86F70D01 01010500 03818D00
  30818902 818100BD F3B837AA D925F391 2B64DA14 9C2EA031 5A7203C4 92F8D6A8
  7D2357A6 BCC8596F A38A9B10 47435626 D59A8F2A 123195BB BE5A1E74 B1AA5AE0
  5CA162FF 8C3ACA4F B3EE9F27 8B031642 B618AE1B 40F2E3B4 F996BEFE 382C7283
  3792A369 236F8561 8748AA3F BC41F012 B859BD9C DB4F75EE 3CEE2829 704BD68F
  FD904043 0F555702 03010001 A3573055 30250603 551D1F04 1E301C30 1AA018A0
  16861468 7474703A 2F2F3633 2E323437 2E313037 2E393330 0B060355 1D0F0404
  030205A0 301F0603 551D2304 18301680 1420FC4B CF0B1C56 F5BD4C06 0AFD4E67
  341AE612 D1300D06 092A8648 86F70D01 01040500 03818100 79E97018 FB955108
  12F42A56 2A6384BC AC8E22FE F1D6187F DA5D6737 C0E241AC AAAEC75D 3C743F59
  08DEEFF2 0E813A73 D79E0FA9 D62DC20D 8E2798CD 2C1DC3EC 3B2505A1 3897330C
  15A60D5A 8A13F06D 51043D37 E56E45DF A65F43D7 4E836093 9689784D C45FD61D
  EC1F160C 1ABC8D03 49FB11B1 DA0BED6C 463E1090 F34C59E4
  quit
 certificate ca 01
  30820207 30820170 A0030201 02020101 300D0609 2A864886 F70D0101 04050030
  17311530 13060355 0403130C 454D2D43 4552542D 53455256 301E170D 30333132
  31363231 34373432 5A170D30 36313231 35323134 3734325A 30173115 30130603
  55040313 0C454D2D 43455254 2D534552 5630819F 300D0609 2A864886 F70D0101
  01050003 818D0030 81890281 8100C14D 833641CF D784F516 DA6B50C0 7B3CB3C9
  589223AB 99A7DC14 04F74EF2 AAEEE8F5 E3BFAE97 F2F980F7 D889E6A1 2C726C69
  54A29870 7E7363FF 3CD1F991 F5A37CFF 3FFDD3D0 9E486C44 A2E34595 C2D078BB
  E9DE981E B733B868 AA8916C0 A8048607 D34B83C0 64BDC101 161FC103 13C06500
  22D6EE75 7D6CF133 7F1B515F 32830203 010001A3 63306130 0F060355 1D130101
  FF040530 030101FF 300E0603 551D0F01 01FF0404 03020186 301D0603 551D0E04
  16041420 FC4BCF0B 1C56F5BD 4C060AFD 4E67341A E612D130 1F060355 1D230418
  30168014 20FC4BCF 0B1C56F5 BD4C060A FD4E6734 1AE612D1 300D0609 2A864886
  F70D0101 04050003 81810085 D2E386F5 4107116B AD3AC990 CBE84063 5FB2A6B5
  BD572026 528E92ED 02F3A0AE 1803F2AE AA4C0ED2 0F59F18D 7B50264F 30442C41
  0AF19C4E 70BD3CB5 0ADD8DE8 8EF636BD 24410DF4 DB62DAFC 67DA6E58 3879AA3E
  12AFB1C3 2E27CB27 EC74E1FC AEE2F5CF AA80B439 615AA8D5 6D6DEDC3 7F9C2C79
  3963E363 F2989FB9 795BA8
  quit
!
!
crypto isakmp policy 10
 encr aes
 group 14
!
!
crypto ipsec transform-set ISC_TS_1 esp-aes esp-sha-hmac
!
crypto ipsec profile ISC_IPSEC_PROFILE_2
 set security-association lifetime kilobytes 530000000
 set security-association lifetime seconds 14400
 set transform-set ISC_TS_1
!
!
controller ISA 1/1
!
!
interface Tunnel0
 description MGRE Interface provisioned by ISC
 bandwidth 10000
 ip address 192.0.2.172 255.255.255.0
```

```
 no ip redirects
 ip mtu 1408
 ip nhrp map multicast dynamic
 ip nhrp network-id 101
 ip nhrp holdtime 500
 ip nhrp server-only
 no ip split-horizon eigrp 101
 tunnel source FastEthernet2/1
 tunnel mode gre multipoint
 tunnel key 101
 tunnel protection ipsec profile ISC_IPSEC_PROFILE_2
!
interface FastEthernet2/0
 ip address 192.0.2.1 255.255.255.0
 duplex auto
 speed auto
!
interface FastEthernet2/1
 ip address 192.0.2.2 255.255.255.0
 duplex auto
 speed auto
!
!
end
```

## Example: Debug of a Successful PKI AAA Authorization

The following **show debugging** command output shows a successful authorization using the PKI Integration with AAA Server feature:

```
Device#show debugging

General OS:
  TACACS access control debugging is on
  AAA Authentication debugging is on
  AAA Authorization debugging is on
Cryptographic Subsystem:
Crypto PKI Trans debugging is on
Device#
May 28 19:36:11.117: CRYPTO_PKI: Trust-Point EM-CERT-SERV picked up
May 28 19:36:12.789: CRYPTO_PKI: Found a issuer match
May 28 19:36:12.805: CRYPTO_PKI: cert revocation status unknown.
May 28 19:36:12.805: CRYPTO_PKI: Certificate validated without revocation check
May 28 19:36:12.813: CRYPTO_PKI_AAA: checking AAA authorization (ACSLab, POD5.example.com,
 <all>)
May 28 19:36:12.813: AAA/BIND(00000042): Bind i/f
May 28 19:36:12.813: AAA/AUTHOR (0x42): Pick method list 'ACSLab'
May 28 19:36:12.813: TPLUS: Queuing AAA Authorization request 66 for processing
May 28 19:36:12.813: TPLUS: processing authorization request id 66
May 28 19:36:12.813: TPLUS: Protocol set to None .....Skipping
May 28 19:36:12.813: TPLUS: Sending AV service=pki
May 28 19:36:12.813: TPLUS: Authorization request created for 66(POD5.example.com)
May 28 19:36:12.813: TPLUS: Using server 192.0.2.55
May 28 19:36:12.813: TPLUS(00000042)/0/NB_WAIT/203A4628: Started 5 sec timeout
May 28 19:36:12.813: TPLUS(00000042)/0/NB_WAIT: wrote entire 46 bytes request
May 28 19:36:12.813: TPLUS: Would block while reading pak header
May 28 19:36:12.817: TPLUS(00000042)/0/READ: read entire 12 header bytes (expect 27 bytes)
May 28 19:36:12.817: TPLUS(00000042)/0/READ: read entire 39 bytes response
May 28 19:36:12.817: TPLUS(00000042)/0/203A4628: Processing the reply packet
May 28 19:36:12.817: TPLUS: Processed AV cert-application=all
May 28 19:36:12.817: TPLUS: received authorization response for 66: PASS
May 28 19:36:12.817: CRYPTO_PKI_AAA: reply attribute ("cert-application" = "all")
May 28 19:36:12.817: CRYPTO_PKI_AAA: authorization passed
```

```
Device#
May 28 19:36:18.681: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 101: Neighbor 192.0.2.171 (Tunnel0) is
 up: new adjacency
Device#
Device# show crypto isakmp sa

dst             src             state          conn-id slot
192.0.2.22     192.0.2.102   QM_IDLE             84    0
```

## Example:Debug of a Failed PKI AAA Authorization

The following **show debugging** command output shows that the device is not authorized to connect using VPN. The messages are typical of those that you might see in such a situation.

In this example, the peer username was configured as not authorized, by moving the username to a Cisco Secure ACS group called VPN_Disabled in Cisco Secure ACS. The device, device9.example.com, has been configured to check with a Cisco Secure ACS AAA server prior to establishing a VPN connection to any peer.

```
Device#show debugging

General OS:
  TACACS access control debugging is on
  AAA Authentication debugging is on
  AAA Authorization debugging is on
Cryptographic Subsystem:
  Crypto PKI Trans debugging is on

Device#
May 28 19:48:29.837: CRYPTO_PKI: Trust-Point EM-CERT-SERV picked up
May 28 19:48:31.509: CRYPTO_PKI: Found a issuer match
May 28 19:48:31.525: CRYPTO_PKI: cert revocation status unknown.
May 28 19:48:31.525: CRYPTO_PKI: Certificate validated without revocation check
May 28 19:48:31.533: CRYPTO_PKI_AAA: checking AAA authorization (ACSLab, POD5.example.com,
 <all>)
May 28 19:48:31.533: AAA/BIND(00000044): Bind i/f
May 28 19:48:31.533: AAA/AUTHOR (0x44): Pick method list 'ACSLab'
May 28 19:48:31.533: TPLUS: Queuing AAA Authorization request 68 for processing
May 28 19:48:31.533: TPLUS: processing authorization request id 68
May 28 19:48:31.533: TPLUS: Protocol set to None .....Skipping
May 28 19:48:31.533: TPLUS: Sending AV service=pki
May 28 19:48:31.533: TPLUS: Authorization request created for 68(POD5.example.com)
May 28 19:48:31.533: TPLUS: Using server 192.0.2.55
May 28 19:48:31.533: TPLUS(00000044)/0/NB_WAIT/203A4C50: Started 5 sec timeout
May 28 19:48:31.533: TPLUS(00000044)/0/NB_WAIT: wrote entire 46 bytes request
May 28 19:48:31.533: TPLUS: Would block while reading pak header
May 28 19:48:31.537: TPLUS(00000044)/0/READ: read entire 12 header bytes (expect 6 bytes)
May 28 19:48:31.537: TPLUS(00000044)/0/READ: read entire 18 bytes response
May 28 19:48:31.537: TPLUS(00000044)/0/203A4C50: Processing the reply packet
May 28 19:48:31.537: TPLUS: received authorization response for 68: FAIL
May 28 19:48:31.537: CRYPTO_PKI_AAA: authorization declined by AAA, or AAA server not found.
May 28 19:48:31.537: CRYPTO_PKI_AAA: No cert-application attribute found. Failing.
May 28 19:48:31.537: CRYPTO_PKI_AAA: authorization failed
May 28 19:48:31.537: CRYPTO_PKI: AAA authorization for list 'ACSLab', and user
'POD5.example.com' failed.
May 28 19:48:31.537: %CRYPTO-5-IKMP_INVAL_CERT: Certificate received from 192.0.2.162 is
bad: certificate invalid
May 28 19:48:39.821: CRYPTO_PKI: Trust-Point EM-CERT-SERV picked up
May 28 19:48:41.481: CRYPTO_PKI: Found a issuer match
May 28 19:48:41.501: CRYPTO_PKI: cert revocation status unknown.
May 28 19:48:41.501: CRYPTO_PKI: Certificate validated without revocation check
```

```
May 28 19:48:41.505: CRYPTO_PKI_AAA: checking AAA authorization (ACSLab, POD5.example.com,
 <all>)
May 28 19:48:41.505: AAA/BIND(00000045): Bind i/f
May 28 19:48:41.505: AAA/AUTHOR (0x45): Pick method list 'ACSLab'
May 28 19:48:41.505: TPLUS: Queuing AAA Authorization request 69 for processing
May 28 19:48:41.505: TPLUS: processing authorization request id 69
May 28 19:48:41.505: TPLUS: Protocol set to None .....Skipping
May 28 19:48:41.505: TPLUS: Sending AV service=pki
May 28 19:48:41.505: TPLUS: Authorization request created for 69(POD5.example.com)
May 28 19:48:41.505: TPLUS: Using server 198.168.244.55
May 28 19:48:41.509: TPLUS(00000045)/0/IDLE/63B22834: got immediate connect on new 0
May 28 19:48:41.509: TPLUS(00000045)/0/WRITE/63B22834: Started 5 sec timeout
May 28 19:48:41.509: TPLUS(00000045)/0/WRITE: wrote entire 46 bytes request
May 28 19:48:41.509: TPLUS(00000045)/0/READ: read entire 12 header bytes (expect 6 bytes)
May 28 19:48:41.509: TPLUS(00000045)/0/READ: read entire 18 bytes response
May 28 19:48:41.509: TPLUS(00000045)/0/63B22834: Processing the reply packet
May 28 19:48:41.509: TPLUS: received authorization response for 69: FAIL
May 28 19:48:41.509: CRYPTO_PKI_AAA: authorization declined by AAA, or AAA server not found.
May 28 19:48:41.509: CRYPTO_PKI_AAA: No cert-application attribute found. Failing.
May 28 19:48:41.509: CRYPTO_PKI_AAA: authorization failed
May 28 19:48:41.509: CRYPTO_PKI: AAA authorization for list 'ACSLab', and user
'POD5.example.com' failed.
May 28 19:48:41.509: %CRYPTO-5-IKMP_INVAL_CERT: Certificate received from 192.0.2.162 is
bad: certificate invalid
Device#
Device# show crypto iskmp sa

dst             src             state          conn-id slot
192.0.2.2        192.0.2.102   MM_KEY_EXCH         95    0
```

## Examples: Configuring a Revocation Mechanism

This section contains the following configuration examples that can be used when specifying a revocation mechanism for your PKI:

### Example:Configuring an OCSP Server

The following example shows how to configure the router to use the OCSP server that is specified in the AIA extension of the certificate:

```
Device> enable
Device# configure terminal
Device(config)#crypto pki trustpoint mytp
Device(ca-trustpoint)# revocation-check ocsp
Device(ca-trustpoint)# end
```

### Example:Specifying a CRL and Then an OCSP Server

The following example shows how to configure the router to download the CRL from the CDP. If the CRL is unavailable, the OCSP server that is specified in the AIA extension of the certificate will be used. If both options fail, certificate verification will also fail.

```
Device> enable
Device# configure terminal
Device(config)#crypto pki trustpoint mytp
Device(ca-trustpoint)#revocation-check crl ocsp
Device(ca-trustpoint)# end
```

### Example: Specifying an OCSP Server

The following example shows how to configure your router to use the OCSP server at the HTTP URL "http://myocspserver:81." If the server is down, the revocation check will be ignored.

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint mytp
Device(ca-trustpoint)# ocsp url http://myocspserver:81
Device(ca-trustpoint)# revocation-check ocsp none
Device(ca-trustpoint)# end
```

### Example: Disabling Nonces in Communications with the OCSP Server

The following example shows communications when a nonce, or a unique identifier for the OCSP request, is disabled for communications with the OCSP server:

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint mytp
Device(ca-trustpoint)# ocsp url http://myocspserver:81
Device(ca-trustpoint)# revocation-check ocsp none
Device(ca-trustpoint)# ocsp disable-nonce
Device(ca-trustpoint)# end
```

## Example:Configuring a Hub Device at a Central Site for Certificate Revocation Checks

The following example shows a hub device at a central site that is providing connectivity for several branch offices to the central site.

The branch offices are also able to communicate directly with each other using additional IPSec tunnels between the branch offices.

The CA publishes CRLs on an HTTP server at the central site. The central site checks CRLs for each peer when setting up an IPSec tunnel with that peer.

The example does not show the IPSec configuration--only the PKI-related configuration is shown.

#### Home Office Hub Configuration

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint VPN-GW
Device(ca-trustpoint)# enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
Device(ca-trustpoint)# serial-number none
Device(ca-trustpoint)# fqdn none
Device(ca-trustpoint)# ip-address none
Device(ca-trustpoint)# subject-name o=Home Office Inc,cn=Central VPN Gateway
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# end
```

#### Central Site Hub Device

```
Device# show crypto ca certificate

Certificate
  Status: Available
```

```
  Certificate Serial Number: 2F62BE14000000000CA0
  Certificate Usage: General Purpose
  Issuer:
    cn=Central Certificate Authority
    o=Home Office Inc
  Subject:
    Name: Central VPN Gateway
    cn=Central VPN Gateway
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 00:43:26 GMT Sep 26 2003
    end   date: 00:53:26 GMT Sep 26 2004
    renew date: 00:00:00 GMT Jan 1 1970
  Associated Trustpoints: VPN-GW
CA Certificate
  Status: Available
  Certificate Serial Number: 1244325DE0369880465F977A18F61CA8
  Certificate Usage: Signature
  Issuer:
    cn=Central Certificate Authority
    o=Home Office Inc
  Subject:
    cn=Central Certificate Authority
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 22:19:29 GMT Oct 31 2002
    end   date: 22:27:27 GMT Oct 31 2017
  Associated Trustpoints: VPN-GW
```

### Trustpoint on the Branch Office Device

```
Device> enable
Device# configure terminal
Device(ca-trustpoint)# crypto pki trustpoint home-office
Device(ca-trustpoint)# enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
Device(ca-trustpoint)# serial-number none
Device(ca-trustpoint)# fqdn none
Device(ca-trustpoint)# ip-address none
Device(ca-trustpoint)# subject-name o=Home Office Inc,cn=Branch 1
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# end
```

A certificate map is entered on the branch office device.

```
branch1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
branch1(config)# crypto pki certificate map central-site 10
branch1(ca-certificate-map)# end
```

The output from the **show certificate** command on the central site hub device shows that the certificate was issued by the following:

```
cn=Central Certificate Authority
o=Home Office Inc
```

These two lines are combined into one line using a comma (,) to separate them, and the original lines are added as the first criteria for a match.

```
Device(ca-certificate-map)# issuer-name co cn=Central Certificate Authority, ou=Home Office
 Inc
!The above line wrapped but should be shown on one line with the line above it.
```

The same combination is done for the subject name from the certificate on the central site device (note that the line that begins with “Name:” is not part of the subject name and must be ignored when creating the certificate map criteria). This is the subject name to be used in the certificate map.

cn=Central VPN Gateway

o=Home Office Inc

```
Device(ca-certificate-map)# subject-name eq cn=central vpn gateway, o=home office inc
```

Now the certificate map is added to the trustpoint that was configured earlier.

```
Device> enable
Device# configure terminal
Device(ca-certificate-map)# crypto pki trustpoint home-office
Device(ca-trustpoint)# match certificate central-site skip revocation-check
Device(ca-trustpoint)# end
```

The configuration is checked (most of configuration is not shown).

```
Device# write term

!Many lines left out
.
.
.
crypto pki trustpoint home-office
 enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
 serial-number none
 fqdn none
 ip-address none
 subject-name o=Home Office Inc,cn=Branch 1
 revocation-check crl
 match certificate central-site skip revocation-check
!
!
crypto pki certificate map central-site 10
 issuer-name co cn = Central Certificate Authority, ou = Home Office Inc
 subject-name eq cn = central vpn gateway, o = home office inc
!many lines left out
```

Note that the issuer-name and subject-name lines have been reformatted to make them consistent for later matching with the certificate of the peer.

If the branch office is checking the AAA, the trustpoint will have lines similar to the following:

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint home-office
Device(ca-trustpoint)# authorization list allow_list
Device(ca-trustpoint)# authorization username subjectname commonname
Device(ca-trustpoint)# end
```

After the certificate map has been defined as was done above, the following command is added to the trustpoint to skip AAA checking for the central site hub.

```
Device(ca-trustpoint)# match certificate central-site skip authorization-check
```

In both cases, the branch site device has to establish an IPSec tunnel to the central site to check CRLs or to contact the AAA server. However, without the **match certificate**command and **central-site skip authorization-check (argument and keyword)**, the branch office cannot establish the tunnel until it has checked the CRL or the AAA server. (The tunnel will not be established unless the **match certificate**command and **central-site skip authorization-check** argument and keyword are used.)

The **match certificate** command and **allow expired-certificate** keyword would be used at the central site if the device at a branch site had an expired certificate and it had to establish a tunnel to the central site to renew its certificate.

### Trustpoint on the Central Site Device

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint VPN-GW
Device(ca-trustpoint)# enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
Device(ca-trustpoint)# serial-number none
Device(ca-trustpoint)# fqdn none
Device(ca-trustpoint)# ip-address none
Device(ca-trustpoint)# subject-name o=Home Office Inc,cn=Central VPN Gateway
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# end
```

### Trustpoint on the Branch 1 Site Device

```
Device# show crypto ca certificate

Certificate
  Status: Available
  Certificate Serial Number: 2F62BE14000000000CA0
  Certificate Usage: General Purpose
  Issuer:
    cn=Central Certificate Authority
    o=Home Office Inc
  Subject:
    Name: Branch 1 Site
    cn=Branch 1 Site
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 00:43:26 GMT Sep 26 2003
    end   date: 00:53:26 GMT Oct 3 2003
    renew date: 00:00:00 GMT Jan 1 1970
  Associated Trustpoints: home-office
CA Certificate
  Status: Available
  Certificate Serial Number: 1244325DE0369880465F977A18F61CA8
  Certificate Usage: Signature
  Issuer:
    cn=Central Certificate Authority
    o=Home Office Inc
  Subject:
    cn=Central Certificate Authority
```

```
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 22:19:29 GMT Oct 31 2002
    end   date: 22:27:27 GMT Oct 31 2017
  Associated Trustpoints: home-office
```

A certificate map is entered on the central site device.

```
Device> enable
Device# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Device(config)# crypto pki certificate map branch1 10
Device(ca-certificate-map)# issuer-name co cn=Central Certificate Authority, ou=Home Office
 Inc
!The above line wrapped but should be part of the line above it.
Device(ca-certificate-map)# subject-name eq cn=Brahcn 1 Site,o=home office inc
Device(ca-certificate-map)# end
```

The certificate map is added to the trustpoint.

```
Device> enable
Device# configure terminal
Device(ca-certificate-map)# crypto pki trustpoint VPN-GW
Device(ca-trustpoint)# match certificate branch1 allow expired-certificate
Device(ca-trustpoint)# exit
Device (config) #exit
```

The configuration should be checked (most of the configuration is not shown).

```
Device# write term

!many lines left out
crypto pki trustpoint VPN-GW
 enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
 serial-number none
 fqdn none
 ip-address none
 subject-name o=Home Office Inc,cn=Central VPN Gateway
 revocation-check crl
 match certificate branch1 allow expired-certificate
!
!
crypto pki certificate map central-site 10
 issuer-name co cn = Central Certificate Authority, ou = Home Office Inc
 subject-name eq cn = central vpn gateway, o = home office inc
! many lines left out
```

The **match certificate**command and **branch1 allow expired-certificate** (argument and keyword) and the certificate map should be removed as soon as the branch device has a new certificate.

## Examples:Configuring Certificate Authorization and Revocation Settings

This section contains the following configuration examples that can be used when specifying a CRL cache control setting or certificate serial number session control:

### Example: Configuring CRL Cache Control

The following example shows how to disable CRL caching for all CRLs associated with the CA1 trustpoint:

```
Device> enable
Device# configure terminal
```

```
Device(config)# crypto pki trustpoint CA1
Device(ca-trustpoint)# enrollment url http://CA1:80
Device(ca-trustpoint)# ip-address FastEthernet0/0
Device(ca-trustpoint)# crl query ldap://ldap_CA1
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# crl cache none
Device(ca-trustpoint)# end
```

The current CRL is still cached immediately after executing the example configuration shown above:

Device# **show crypto pki crls**

```
CRL Issuer Name:
    cn=name Cert Manager,ou=pki,o=example.com,c=US
    LastUpdate: 18:57:42 GMT Nov 26 2005
    NextUpdate: 22:57:42 GMT Nov 26 2005
    Retrieved from CRL Distribution Point:
      ldap://ldap.example.com/CN=name Cert Manager,O=example.com
```

When the current CRL expires, a new CRL is then downloaded to the router at the next update. The **crl-cache none**command takes effect and all CRLs for the trustpoint are no longer cached; caching is disabled. You can verify that no CRL is cached by executing the **show crypto pki crls** command. No output will be shown because there are no CRLs cached.

The following example shows how to configure the maximum lifetime of 2 minutes for all CRLs associated with the CA1 trustpoint:

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint CA1
Device(ca-trustpoint)# enrollment url http://CA1:80
Device(ca-trustpoint)# ip-address FastEthernet 0/0
Device(ca-trustpoint)# crl query ldap://ldap_CA1
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# crl cache delete-after 2
Device(ca-trustpoint)# end
```

The current CRL is still cached immediately after executing the example configuration above for setting the maximum lifetime of a CRL:

Device# **show crypto pki crls**

```
CRL Issuer Name:
    cn=name Cert Manager,ou=pki,o=example.com,c=US
    LastUpdate: 18:57:42 GMT Nov 26 2005
    NextUpdate: 22:57:42 GMT Nov 26 2005
    Retrieved from CRL Distribution Point:
      ldap://ldap.example.com/CN=name Cert Manager,O=example.com
When the current CRL expires, a new CRL is downloaded to the router at the next update and
 the crl-cache delete-after
command takes effect. This newly cached CRL and all subsequent CRLs will be deleted after
a maximum lifetime of 2 minutes.
You can verify that the CRL will be cached for 2 minutes by executing the show crypto pki
crls
 command. Note that the NextUpdate time is 2 minutes after the LastUpdate time.
```

Device# **show crypto pki crls**

```
CRL Issuer Name:
    cn=name Cert Manager,ou=pki,o=example.com,c=US
    LastUpdate: 22:57:42 GMT Nov 26 2005
```

```
    NextUpdate: 22:59:42 GMT Nov 26 2005
    Retrieved from CRL Distribution Point:
```

ldap://ldap.example.com/CN=name Cert Manager,O=example.com

## Example: Configuring Certificate Serial Number Session Control

The following example shows the configuration of certificate serial number session control using a certificate map for the CA1 trustpoint:

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint CA1
Device(ca-trustpoint)# enrollment url http://CA1
Device(ca-trustpoint)# chain-validation stop
Device(ca-trustpoint)# crl query ldap://ldap_server
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# match certificate crl
Device(ca-trustpoint)# exit
Device(config)# crypto pki certificate map crl 10
Device(ca-certificate-map)# serial-number co 279d
Device(ca-certificate-map)# end
```

**Note** If the *match-criteria* value is set to **eq** (equal) instead of **co** (contains), the serial number must match the certificate map serial number exactly, including any spaces.

The following example shows the configuration of certificate serial number session control using AAA attributes. In this case, all valid certificates will be accepted if the certificate does not have the serial number "4ACA."

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint CA1
Device(ca-trustpoint)# enrollment url http://CA1
Device(ca-trustpoint)# ip-address FastEthernet0/0
Device(ca-trustpoint)# crl query ldap://ldap_CA1
Device(ca-trustpoint)# revocation-check crl
Device(ca-trustpoint)# exit
Device(config)# aaa new-model
Device(config)# aaa attribute list crl
Device(config-attr-list)# attribute-type aaa-cert-serial-not 4ACA
Device(config-attr-list)# end
```

The server log shows that the certificate with the serial number "4ACA" was rejected. The certificate rejection is shown using exclamation points.

```
.
.
.
Dec 3 04:24:39.051: CRYPTO_PKI: Trust-Point CA1 picked up
Dec 3 04:24:39.051: CRYPTO_PKI: locked trustpoint CA1, refcount is 1
Dec 3 04:24:39.051: CRYPTO_PKI: unlocked trustpoint CA1, refcount is 0
Dec 3 04:24:39.051: CRYPTO_PKI: locked trustpoint CA1, refcount is 1
Dec 3 04:24:39.135: CRYPTO_PKI: validation path has 1 certs
Dec 3 04:24:39.135: CRYPTO_PKI: Found a issuer match
Dec 3 04:24:39.135: CRYPTO_PKI: Using CA1 to validate certificate
Dec 3 04:24:39.135: CRYPTO_PKI: Certificate validated without revocation check
```

```
Dec 3 04:24:39.135: CRYPTO_PKI: Selected AAA username: 'PKIAAA'
Dec 3 04:24:39.135: CRYPTO_PKI: Anticipate checking AAA list:'CRL'
Dec 3 04:24:39.135: CRYPTO_PKI_AAA: checking AAA authorization (CRL, PKIAAA-L1, <all>)
Dec 3 04:24:39.135: CRYPTO_PKI_AAA: pre-authorization chain validation status (0x4)
Dec 3 04:24:39.135: AAA/BIND(00000021): Bind i/f
Dec 3 04:24:39.135: AAA/AUTHOR (0x21): Pick method list 'CRL'
.
.
.
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: reply attribute ("cert-application" = "all")
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: reply attribute ("cert-trustpoint" = "CA1")
!
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: reply attribute ("cert-serial-not" = "4ACA")
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: cert-serial doesn't match ("4ACA" != "4ACA")
!
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: post-authorization chain validation status (0x7)
!
Dec 3 04:24:39.175: CRYPTO_PKI: AAA authorization for list 'CRL', and user 'PKIAAA' failed.
Dec 3 04:24:39.175: CRYPTO_PKI: chain cert was anchored to trustpoint CA1, and chain
validation result was:
 CRYPTO_PKI_CERT_NOT_AUTHORIZED
!
Dec 3 04:24:39.175: %CRYPTO-5-IKMP_INVAL_CERT: Certificate received from 192.0.2.43 is bad:
 certificate invalid
Dec 3 04:24:39.175: %CRYPTO-6-IKMP_MODE_FAILURE: Processing of Main mode failed with peer
at 192.0.2.43
.
.
.
```

## Examples: Configuring Certificate Chain Validation

This section contains the following configuration examples that can be used to specify the level of certificate chain processing for your device certificates:

### Configuring Certificate Chain Validation from Peer to Root CA

In the following configuration example, all of the certificates will be validated--the peer, SubCA11, SubCA1, and RootCA certificates.

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint RootCA
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation stop
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair RootCA
Device(ca-trustpoint)# exit
Device(config)# crypto pki trustpoint SubCA1
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation continue RootCA
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair SubCA1
Device(ca-trustpoint)# exit
Device(config)# crypto pki trustpoint SubCA11
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation continue SubCA1
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair SubCA11
Device(ca-trustpoint)# end
```

### Configuring Certificate Chain Validation from Peer to Subordinate CA

In the following configuration example, the following certificates will be validated--the peer and SubCA1 certificates.

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint RootCA
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation stop
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair RootCA
Device(ca-trustpoint)# exit
Device(config)# crypto pki trustpoint SubCA1
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation continue RootCA
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair SubCA1
Device(ca-trustpoint)# exit
Device(config)# crypto pki trustpoint SubCA11
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation continue SubCA1
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair SubCA11
Device(ca-trustpoint)# end
```

### Configuring Certificate Chain Validation Through a Gap

In the following configuration example, SubCA1 is not in the configured Cisco IOS hierarchy but is expected to have been supplied in the certificate chain presented by the peer.

If the peer supplies the SubCA1 certificate in the presented certificate chain, the following certificates will be validated--the peer, SubCA11, and SubCA1 certificates.

If the peer does not supply the SubCA1 certificate in the presented certificate chain, the chain validation will fail.

```
Device> enable
Device# configure terminal
Device(config)# crypto pki trustpoint RootCA
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation stop
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair RootCA
Device(ca-trustpoint)# exit
Device(config)# crypto pki trustpoint SubCA11
Device(ca-trustpoint)# enrollment terminal
Device(ca-trustpoint)# chain-validation continue RootCA
Device(ca-trustpoint)# revocation-check none
Device(ca-trustpoint)# rsakeypair SubCA11
Device(ca-trustpoint)# end
```

## Feature History for Authorization and Revocation of Certificates in a PKI

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Fuji 16.8.1a | Authorization and Revocation of Certificates in a PKI | Certificates contain several fields that are used to determine whether a device or user is authorized to perform a specified action. Certificate-based ACLs also help determine when PKI components such as revocation, authorization, or a trustpoint should be used.<br><br>Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 47

# Source Interface Selection for Outgoing Traffic with Certificate Authority

The Source Interface Selection for Outgoing Traffic with Certificate Authority feature allows the IP address of an interface to be specified and used as the source address for all outgoing TCP connections associated with that trustpoint when a designated trustpoint has been configured.


- Information About Source Interface Selection for Outgoing Traffic with Certificate Authority, on page 829
- How to Configure Source Interface Selection for Outgoing Traffic with Certificate Authority, on page 830
- Configuration Examples for Source Interface Selection for Outgoing Traffic with Certificate Authority, on page 833
- Feature History for Source Interface Selection for Outgoing Traffic with Certificate Authority, on page 833


# Information About Source Interface Selection for Outgoing Traffic with Certificate Authority

## Certificates That Identify an Entity

Certificates can be used to identify an entity. A trusted server, known as the certification authority (CA), issues the certificate to the entity after determining the identity of the entity. A device that is running Cisco IOS XE software obtains its certificate by making a network connection to the CA. Using the Simple Certificate Enrollment Protocol (SCEP), the device transmits its certificate request to the CA and receives the granted certificate. The device obtains the certificate of the CA in the same manner using SCEP. When validating a certificate from a remote device, the device may again contact the CA or a Lightweight Directory Access Protocol (LDAP) or HTTP server to determine whether the certificate of the remote device has been revoked. (This process is known as checking the certificate revocation list [CRL].)

In some configurations, the device may make the outgoing TCP connection using an interface that does not have a valid or IP address that can be routed. The user must specify that the address of a different interface be used as the source IP address for the outgoing connection. Cable modems are a specific example of this requirement because the outgoing cable interface (the RF interface) usually does not have an IP address that can be routed. However, the user interface (usually Ethernet) does have a valid IP address.

# Source Interface for Outgoing TCP Connections Associated with a Trustpoint

The **crypto ca trustpoint** command is used to specify a trustpoint. The **source interface**command is used along with the **crypto ca trustpoint**command to specify the address of the interface that is to be used as the source address for all outgoing TCP connections associated with that trustpoint.

**Note** If the interface address is not specified using the **source interface**command, the address of the outgoing interface is used.

# How to Configure Source Interface Selection for Outgoing Traffic with Certificate Authority

## Configuring the Interface for All Outgoing TCP Connections Associated with a Trustpoint

Perform this task to configure the interface that you want to use as the source address for all outgoing TCP connections associated with a trustpoint.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **crypto ca trustpoint** *name*<br>**Example:**<br>`Device(config)# crypto ca trustpoint ms-ca` | Declares the Certificate Authority (CA) that your device should use and enters ca-trustpoint configuration mode. |
| **Step 4** | **enrollment** [**mode**] [**retry period** *minutes*] [**retry count** *number*] **url** *url* [**pem**]<br>**Example:** | Specifies the following enrollment parameters of the CA:<br>• (Optional) The **mode** keyword specifies the registration authority (RA) mode, if |

| | Command or Action | Purpose |
|---|---|---|
| | `Device(ca-trustpoint)# enrollment url http://caserver.myexample.com`<br>- or-<br>`Device(ca-trustpoint)# enrollment url http://[2001:DB8:1:1::1]:80` | your CA system provides an RA. By default, RA mode is disabled.<br>• (Optional) The **retry period** keyword and *minutes* argument specifies the period, in minutes, in which the device waits before sending the CA another certificate request. Valid values are from 1 to 60. The default is 1.<br>• (Optional) The **retry count** keyword and *number* argument specifies the number of times a device will resend a certificate request when it does not receive a response from the previous request. Valid values are from 1 to 100. The default is 10.<br>• The *url* argument is the URL of the CA to which your device should send certificate requests.<br>• (Optional) The **pem** keyword adds privacy-enhanced mail (PEM) boundaries to the certificate request. |
| **Step 5** | **source interface** *interface-address*<br>**Example:**<br>`Device(ca-trustpoint)# source interface gigabitethernet 0/1/0` | Interface to be used as the source address for all outgoing TCP connections associated with that trustpoint. |
| **Step 6** | **exit**<br>**Example:**<br>`Device(ca-trustpoint)# exit` | Exits ca-trustpoint configuration mode and returns to global configuration mode. |
| **Step 7** | **interface** *type slot / port*<br>**Example:**<br>`Device(config)# interface gigabitethernet 1/0/1` | Configures an interface type and enters interface configuration mode. |
| **Step 8** | **description** *string*<br>**Example:**<br>`Device(config-if)# description inside interface` | Adds a description to an interface configuration. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | **ip address** *ip-address mask*<br>**Example:**<br>`Device(config-if)# ip address 10.1.1.1 255.255.255.0` | Sets a primary or secondary IP address for an interface. |
| **Step 10** | **exit**<br>**Example:**<br>`Device(config-if)# exit` | Exits interface configuration mode and returns to global configuration mode. |
| **Step 11** | **interface** *type slot/port*<br>**Example:**<br>`Device(config-if)# interface gigabitethernet 1/0/2` | Configures an interface and enters interface configuration mode. |
| **Step 12** | **description** *string*<br>**Example:**<br>`Device(config-if)# description outside interface 10.1.1.205 255.255.255.0` | Adds a description to an interface configuration. |
| **Step 13** | **ip address** *ip-address mask*<br>**Example:**<br>`Device(config-if)# ip address 10.2.2.205 255.255.255.0` | Sets a primary or secondary IP address for an interface. |
| **Step 14** | **crypto map** *map-name*<br>**Example:**<br>`Device(config-if)# crypto map mymap` | Applies a previously defined crypto map set to an interface and enters crypto map configuration mode. |
| **Step 15** | **end**<br>**Example:**<br>`Device(config-crypto-map)# end` | Exits crypto map configuration mode and returns to privileged EXEC mode. |

# Configuration Examples for Source Interface Selection for Outgoing Traffic with Certificate Authority

## Example:Source Interface Selection for Outgoing Traffic with Certificate Authority

In the following example, the device is located in a branch office. The device uses IPSec to communicate with the main office.GigabitEthernet 1/0/1 is the “outside” interface that connects to the ISP. GigabitEthernet 0/1/0 is the interface connected to the LAN of the branch office. To access the CA server located in the main office, the device must send its IP datagrams out interface Ethernet 1 (address 10.2.2.205) using the IPSec tunnel. Address 10.2.2.205 is assigned by the ISP. Address 10.2.2.205 is not a part of the branch office or main office.

The CA cannot access any address outside the company because of a firewall. The CA sees a message coming from 10.2.2.205 and cannot respond (that is, the CA does not know that the device is located in a branch office at address 10.1.1.1, which it is able to reach).

Adding the **source interface** command tells the device to use address 10.1.1.1 as the source address of the IP datagram that it sends to the CA. The CA is able to respond to 10.1.1.1.

This scenario is configured using the **source interface** command and the interface addresses as described above.

```
Device> enable
Device# configure terminal
Device(config)# crypto ca trustpoint ms-ca
Device(ca-trustpoint)# enrollment url http://ms-ca:80/certsrv/mscep/mscep.dll
Device(ca-trustpoint)# source interface gigabitethernet 0/1/0
Device(ca-trustpoint)# exit
Device(onfig)# interface gigabitethernet 0/1/0
Device(config-if)# description inside interface
Device(config-if)# ip address 10.1.1.1 255.255.255.0
Device(config-if)# exit
Device(config)# interface gigabitethernet 1/0/1
Device(config-if)# description outside interface
Device(config-if)# ip address 10.2.2.205 255.255.255.0
Device(config-if)# crypto map main-office
Device(config-if)# end
```

# Feature History for Source Interface Selection for Outgoing Traffic with Certificate Authority

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Fuji 16.8.1a | Source Interface Selection for Outgoing Traffic with Certificate Authority | The Source Interface Selection for Outgoing Traffic with Certificate Authority feature allows you to specify that the address of an interface be used as the source address for all outgoing TCP connections associated with that trustpoint when a designated trustpoint has been configured.<br><br>Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 48

# Source Interface and VRF Support in LDAP

The Source Interface and VRF Support in LDAP feature allows you to configure a dedicated LDAP source interface IP address and virtual routing and forwarding (VRF) details on Cisco Integrated Services Routers (ISR) Generation 2. The source interface address (the address can be an IPv4 or IPv6 address) and VRF details are populated while creating a TCP connection between the Cisco ISR Generation 2 and the LDAP server. This module describes how to configure this feature.


- Information About Source Interface and VRF Support in LDAP, on page 835
- How to Configure Source Interface and VRF Support in LDAP, on page 836
- Configuration Examples for Source Interface and VRF Support in LDAP, on page 837
- Feature History for Source Interface and VRF Support in LDAP, on page 837


# Information About Source Interface and VRF Support in LDAP

## Source Interface and VRF Support in LDAP Overview

Prior to the introduction of the Source Interface and VRF Support in LDAP feature, the source interface address cannot be specified in the source IP field of the Lightweight Directory Access Protocol (LDAP) query; instead the tunnel interface IP address was used in the source IP field.

The Source Interface and VRF Support in LDAP feature helps you configure a dedicated LDAP source interface address on a Cisco device. The source interface address is configured on the Cisco device, and the device uses this interface address to originate all LDAP packets it sends to the LDAP server. The source interface address is also used for polling the end-server to ensure the reachability of the end-server.

The source interface IP (either an IPv4 or IPv6 address) address and virtual routing and forwarding (VRF) details are populated in the LDAP query while creating a TCP connection between the Cisco device (client) and the LDAP server.

The VRF instance is configured on the Cisco device and VRF table ID details are set in the socket option before creating a TCP connection to allow multiple instances of a routing table to coexist on the same device at the same time. Because routing instances are independent of each other, the same or overlapping IP address can be used without conflict.

# How to Configure Source Interface and VRF Support in LDAP

## Configuring LDAP Source Interface and VRF

If you have configured the source interface address and virtual routing and forwarding (VRF) instance under the **aaa group server ldap** command and in global configuration mode, the configuration under the **aaa group server ldap** command has the highest priority.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device(config)# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables the authentication, authorization, and accounting (AAA) access control model. |
| **Step 4** | **aaa group server ldap** *group-name*<br>**Example:**<br>`Device(config)# aaa group server ldap ldap-server-group` | Groups different Lightweight Directory Access Protocol (LDAP) servers into distinct lists and methods and enters LDAP server-group configuration mode. |
| **Step 5** | {**ip** \| **ipv6**} **ldap source-interface** *interface-type interface-number*<br>**Example:**<br>`Device(config-ldap-sg)# ip ldap source-interface gigabitethernet 0/0/0` | Specifies the source interface IP address in the LDAP packets. |
| **Step 6** | {**ip** \| **ipv6**} **vrf forwarding** *vrf-name*<br>**Example:**<br>`Device(config-ldap-sg)# ip vrf forwarding cws-vrf` | Configures a VRF reference of a AAA LDAP server group. |
| **Step 7** | **server** *name*<br>**Example:**<br>`Device(config-ldap-sg)# server ldap-server` | Specifies the LDAP server. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | **exit**<br>**Example:**<br>`Device(config-ldap-sg)# exit` | Exits LDAP server-group configuration mode and returns to global configuration mode. |
| **Step 9** | {**ip** \| **ipv6**} **ldap source-interface** *interface-type interface-number* [**vrf** *vrf-name*]<br>**Example:**<br>`Device(config)# ip ldap source-interface gigabitethernet 0/1/0 vrf cws-vrf-1` | Specifies the source interface IP address in the LDAP packets. |
| **Step 10** | **end**<br>**Example:**<br>`Device(config)# end` | Exits global configuration mode and returns to privileged EXEC mode. |

# Configuration Examples for Source Interface and VRF Support in LDAP

## Example: Configuring LDAP Source Interface and VRF

```
Device> enable
Device(config)# configure terminal
Device(config)# aaa new-model
Device(config)# aaa group server ldap ldap-server-group
Device(config-ldap-sg)# ip ldap source-interface gigabitethernet 0/0/0
Device(config-ldap-sg)# ip vrf forwarding cws-vrf
Device(config-ldap-sg)# server ldap-server
Device(config-ldap-sg)# exit
Device(config)# ip ldap source-interface gigabitethernet 0/1/0 vrf cws-vrf-1
Device(config)# end
```

# Feature History for Source Interface and VRF Support in LDAP

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Everest 16.5.1a | Source Interface and VRF Support in LDAP | The source interface, which can be an IPv4 or IPv6 interface, and virtual routing and forwarding (VRF) details are populated while creating a TCP connection between a Cisco device and the LDAP server.<br>Support for this feature was introduced on all the models of the Cisco Catalyst 9500 Series Switches. |
| Cisco IOS XE Fuji 16.8.1a | Source Interface and VRF Support in LDAP | Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 49

# Configuring IPv6 Support for LDAP


- Restrictions for Configuring IPv6 Support for LDAP, on page 839
- Information About Configuring IPv6 Support for LDAP, on page 839
- LDAP Operations, on page 840
- How to Configure IPv6 Support for LDAP, on page 841
- Configuration Examples of IPv6 Support for LDAP, on page 846
- Additional References, on page 847
- Feature History for IPv6 Support for LDAP, on page 848


# Restrictions for Configuring IPv6 Support for LDAP

- Only bind, search, and compare operations are supported.
- The Lightweight Directory Access Protocol (LDAP) referrals are not supported.
- Unsolicited messages or notifications from LDAP server are not handled.

# Information About Configuring IPv6 Support for LDAP

## IPv6 Support for LDAP

To support Lightweight Directory Access Protocol (LDAP) over IPv6, changes are made to authentication, authorization and accounting (AAA) transactions in terms of authentication and authorization while communicating over an IPv6 network. In order to support LDAP over an IPv6 network, transport calls have been modified to support both IPv4 and IPv6 based on the server configuration.

## Transport Layer Security

Transport Layer Security (TLS) is an application-level protocol that enables secure transactions of data through privacy, authentication, and data integrity. It relies upon certificates, public keys, and private keys for clients to prove the identity. Certificates are issued by Certificate Authorities (CAs). Each certificate includes the name of the authority that issued it, the name of the entity to which the certificate was issued, the entity's public key, and time stamps that indicate the certificate's expiration date. TLS support for LDAP is mentioned in RFC 2830 as an extension to the LDAP protocol.

# LDAP Operations

## Bind

The bind operation is used to authenticate a user to the server. It is used to start a connection with the LDAP server. LDAP is a connection-oriented protocol. The client specifies the protocol version and the client authentication information. LDAP supports the following binds:

- Authenticated bind
- Anonymous bind

An authenticated bind is performed when a root distinguished name (DN) and password are available. In the absence of a root DN and password, an anonymous bind is performed. In LDAP deployments, the search operation is performed first and the bind operation later. This is because, if a password attribute is returned as part of the search operation, the password verification can be done locally on an LDAP client. Thus, there is no need to perform an extra bind operation. If a password attribute is not returned, the bind operation can be performed later. Another advantage of performing a search operation first and a bind operation later is that the DN received in the search result can be used as the user DN instead of forming a DN by prefixing the username (cn attribute) with the base DN. All entries stored in an LDAP server have a unique DN. The DN consists of two parts: the Relative Distinguished Name (RDN) and the location within the LDAP server where the record resides.

Most of the entries that you store in an LDAP server will have a name, and the name is frequently stored in the Common Name (cn) attribute. Because every object has a name, most objects you store in an LDAP will use their cn value as the basis for their RDN.

## Compare

The compare operation is used to replace a bind request with a compare request for an authentication. The compare operation helps to maintain the initial bind parameters for the connection.

## Search

A search operation is used to search the LDAP server. The client specifies the starting point (base DN) of the search, the search scope (either the object, its children, or the subtree rooted at the object), and a search filter.

For authorization requests, the search operation is directly performed without a bind operation. The LDAP server can be configured with certain privileges for the search operation to succeed. This privilege level is established with the bind operation.

An LDAP search operation can return multiple user entries for a specific user. In such cases, the LDAP client returns an appropriate error code to AAA. To avoid these errors, appropriate search filters that help to match a single entry must be configured.

# How to Configure IPv6 Support for LDAP

## Configuring Device-to-LDAP Server Communication

The Lightweight Directory Access Protocol (LDAP) host is a multiuser system running LDAP server software, such as Active Directory (Microsoft) and OpenLDAP. Configuring device-to-LDAP server communication can have several components:

- Hostname or IP address
- Port number
- Timeout period
- Base distinguished name (DN)

To configuring Device-to-LDAP server communication, perform this procedure:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **ldap server** *name*<br>**Example:**<br>`Device(config)# ldap server server1` | Configures a device as an LDAP protocol and enters LDAP server configuration mode. |
| **Step 5** | **ipv6** *ipv6-address*<br>**Example:**<br>`Device(config-ldap-server)# ipv6 2001:DB8:0:0:8:800` | Specifies an IPv6 address to the LDAP server. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **transport port** *port-number*<br>**Example:**<br>`Device(config-ldap-server)# transport port 200` | Configures the transport protocol for connecting to the LDAP server. |
| **Step 7** | **timeout retransmit** *seconds*<br>**Example:**<br>`Device(config-ldap-server)# timeout retransmit 20` | Specifies the number of seconds a device waits for a reply to an LDAP request before retransmitting the request. |
| **Step 8** | **exit**<br>**Example:**<br>`Device(config-ldap-server)# exit` | Exits the LDAP server configuration mode and enters global configuration mode. |

# Configuring LDAP Protocol Parameters

To configure LDAP protocol parameters, perform this procedure:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Device> enable` | Enables privileged EXEC mode.<br>Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **ldap server** *name*<br>**Example:**<br>`Device(config)# ldap server server1` | Defines a Lightweight Directory Access Protocol (LDAP) server and enters LDAP server configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **bind authenticate root-dn password** [**0** *string* \| **7** *string*] *string*<br>**Example:**<br>`Device(config-ldap-server)# bind authenticate root-dn "cn=administrator,cn=users,dc=nac-blr2,dc=example,dc=com password"` | Specifies a shared secret text string used between the device and an LDAP server. Use the **0** line option to configure an unencrypted shared secret. Use the **7** line option to configure an encrypted shared secret. |
| **Step 6** | **search-filter user-object-type** *string*<br>**Example:**<br>`Device(config-ldap-server)# search-filter user-object-type string1` | Specifies the search filter to be used in the search requests. |
| **Step 7** | **base-dn** *string*<br>**Example:**<br>`Device(config-ldap-server)# base-dn "dc=sns,dc=example,dc=com"` | Specifies the base distinguished name (DN) of the search. |
| **Step 8** | **mode secure** [**no-negotiation**]<br>**Example:**<br>`Device(config-ldap-server)# mode secure no-negotiation` | Configures LDAP to initiate the transport layer security (TLS) connection and specifies the secure mode. |
| **Step 9** | **secure cipher 3des-ede-cbc-sha**<br>**Example:**<br>`Device(config-ldap-server)# secure cipher 3des-ede-cbc-sha` | Specifies the ciphersuite in the case of a secure connection. |
| **Step 10** | **exit**<br>**Example:**<br>`Device(config-ldap-server)# exit` | Exits LDAP server configuration mode and enters global configuration mode. |

# Configuring Search and Bind Operations for an Authentication Request

To configure search and bind operations for an authentication request, perform this procedure:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable** | Enables privileged EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Device> enable` | Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Device# configure terminal` | Enters global configuration mode. |
| **Step 3** | **aaa new-model**<br>**Example:**<br>`Device(config)# aaa new-model` | Enables AAA. |
| **Step 4** | **ldap server** *name*<br>**Example:**<br>`Device(config)# ldap server server1` | Defines a Lightweight Directory Access Protocol (LDAP) server and enter LDAP server configuration mode. |
| **Step 5** | **authentication bind-first**<br>**Example:**<br>`Device(config-ldap-server)# authentication bind-first` | Configures the sequence of search and bind operations for an authentication request. |
| **Step 6** | **authentication compare**<br>**Example:**<br>`Device(config-ldap-server)# authentication compare` | Replaces the bind request with the compare request for authentication. |
| **Step 7** | **exit**<br>**Example:**<br>`Device(config-ldap-server)# exit` | Exits LDAP server configuration mode. |

# Monitoring and Maintaining LDAP Scalability Enhancements

The following **show** and **debug** commands can be entered in any order.

### Procedure

**Step 1** **enable**

**Example:**

```
> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

**Step 2** **configure terminal**

**Example:**

```
# configure terminal
```

Enters global configuration mode.

**Step 3** **clear ldap server**

Clears the Lightweight Directory Access Protocol (LDAP) server of the TCP connection.

**Example:**

```
# clear ldap server
```

**Step 4** **debug ldap**

Displays information associated with LDAP.

**Example:**

```
# debug ldap
```

**Step 5** **show ldap server**

Displays the LDAP server state information and various other counters for the server.

**Example:**

```
# show ldap server
```

**Step 6** **show ldap attributes**

Displays information about default LDAP attribute mapping.

**Example:**

```
Device# show ldap attributes

LDAP Attribute                    Format      AAA Attribute
==============                    ======      =============
airespaceBwDataBurstContract      Ulong       bsn-data-bandwidth-burst-contr
userPassword                      String      password
airespaceBwRealBurstContract      Ulong       bsn-realtime-bandwidth-burst-c
employeeType                      String      employee-type
airespaceServiceType              Ulong       service-type
airespaceACLName                  String      bsn-acl-name
priv-lvl                          Ulong       priv-lvl
memberOf                          String DN   supplicant-group
cn                                String      username
airespaceDSCP                     Ulong       bsn-dscp
policyTag                         String      tag-name
airespaceQOSLevel                 Ulong       bsn-qos-level
airespace8021PType                Ulong       bsn-8021p-type
airespaceBwRealAveContract        Ulong       bsn-realtime-bandwidth-average
```

```
airespaceVlanInterfaceName       String      bsn-vlan-interface-name
airespaceVapId                   Ulong       bsn-wlan-id
airespaceBwDataAveContract       Ulong       bsn-data-bandwidth-average-con
sAMAccountName                   String      sam-account-name
meetingContactInfo               String      contact-info
telephoneNumber                  String      telephone-number
Map: att_map_1
department                       String DN   element-req-qos
```

# Configuration Examples of IPv6 Support for LDAP

## Example: Device-to-LDAP Server Communication

The following example shows how to create server group server1 and specify the IP address, transport port 200, and retransmit values:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# ldap server server1
Device(config-ldap-server)# ipv6 2001:DB8:0:0:8:800
Device(config-ldap-server)# transport port 200
Device(config-ldap-server)# timeout retransmit 20
Device(config-ldap-server)# exit
```

## Example: LDAP Protocol Parameters

The following example shows how to configure Lightweight Directory Access Protocol (LDAP) parameters:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# ldap server server1
Device(config-ldap-server)# bind authenticate root-dn
"cn=administrator,cn=users,dc=nac-blr2,dc=example,dc=com password"
Device(config-ldap-server)# base-dn "dc=sns,dc=example,dc=com"
Device(config-ldap-server)# mode secure no-negotiation
Device(config-ldap-server)# secure cipher 3des-ede-cbc-sha
Device(config-ldap-server)# exit
```

## Example: Search and Bind Operations for an Authentication Request

The following example shows how to configure the sequence of search and bind operations for an authentication request:

```
Device> enable
Device# configure terminal
Device(config)# aaa new-model
Device(config)# ldap server server1
Device(config-ldap-server)# authentication bind-first
```

```
Device(config-ldap-server)# authentication compare
Device(config-ldap-server)# exit
```

# Example: Server Information from an LDAP Server

The following is sample output from an LDAP server:

```
Device# show ldap server all

Server Information for server1
===============================================
 Server name              :server1
 Server IP                :2001:DB8:0:0:8:800
 Server listening Port    :389
 Connection status        :DOWN
 Root Bind status         :No Bind
 Server mode              :Non-Secure
 Cipher Suite             :0x00
 Authentication Seq       :Search first. Then Bind/Compare      password next
 Authentication Procedure :Bind with user password
 Request timeout          :30
 ---------------------------------------------
 * LDAP STATISTICS *
 Total messages  [Sent:0, Received:0]
 Response delay(ms) [Average:0, Maximum:0]
 Total search    [Request:0, ResultEntry:0, ResultDone:0]
 Total bind      [Request:0, Response:0]
 Total extended  [Request:0, Response:0]
 Total compare   [Request:0, Response:0]
 Search [Success:0, Failures:0]
 Bind   [Success:0, Failures:0]
 Missing attrs in Entry [0]
---------------------------------------------
```

# Additional References

### Related Documents

| Related Topic | Document Title |
|---|---|
| For complete syntax and usage information for the commands used in this chapter. | *Command Reference (Catalyst 9500 Series Switches)* |

### Standards and RFCs

| Standard/RFC | Title |
|---|---|
| RFC 4511 | *Lightweight Directory Access Protocol (LDAP)* |
| RFC 4513 | *Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms* |

# Feature History for IPv6 Support for LDAP

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

| Release | Feature | Feature Information |
|---|---|---|
| Cisco IOS XE Fuji 16.8.1a | IPv6 Support for LDAP | The IPv6 Support for LDAP feature describes IPv6 transport support for the LDAP protocol by introducing changes in authentication, authorization, and accounting (AAA) transactions.<br><br>Support for this feature was introduced on the C9500-32C, C9500-32QC, C9500-48Y4C, and C9500-24Y4C models of the Cisco Catalyst 9500 Series Switches. |

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

CHAPTER 50

# Secure Operation in FIPS Mode


- FIPS 140-2 Overview, on page 849
- Configure FIPS 140-2, on page 850
- Key Zeroization, on page 850
- Disable FIPS Mode, on page 851
- Verify FIPS Configuration, on page 851
- Additional References for Secure Operation in FIPS Mode , on page 852


## FIPS 140-2 Overview

The Federal Information Processing Standards (FIPS) Publication 140-2 (Security Requirements for Cryptographic Modules) details the U.S and Canadian governments' requirements for cryptographic modules. FIPS 140-2 specifies certain cryptographic algorithms as secure, and it also identifies which algorithms should be used if a cryptographic module is to be called FIPS compliant. For more information on the FIPS 140-2 standard and validation program, refer National Institute of Standards and Technology (NIST) website.

The FIPS 140-2 Compliance Review (CR) documents for Cisco Catalyst series switches are posted on the following website:

https://www.cisco.com/c/en/us/solutions/industries/government/global-government-certifications/fips-140.html

Click the link in the "Certification Date" column to view the CR Certificate.

Security Policy document describes the FIPS implementation, hardware installation, firmware initialization, and software configuration procedures for FIPS operation. You can access the FIPS 140-2 Consolidated Validation Certificate and Security Policy document on NIST Computer Security Resource Center. This website opens a Search window. In the **Vendor** field, enter "Cisco" and click **Search**. The resulting window provides a list of Cisco platforms that are FIPS Compliant. From the list, click the desired platform to obtain its Security Policy and Consolidated Certificate.

**Important** This document describes FIPS mode behavior for Cisco Catalyst Switches in general. For more information on platform-specific FIPS 140-2 implementation, refer the FIPS 14-2 Security Policy document for the platform.

# Configure FIPS 140-2

Following is a generic procedure to enable FIPS mode of operation for Cisco Catalyst Switches. For a detailed configuration procedure, refer FIPS 140-2 Security Policy document for the required device.

### Procedure

**Step 1** (Optional) Enable FIPS 140-2 logging.

**Example:**

```
Device(config)# logging console errors
```

**Step 2** Configure Authorization key.

**Example:**

```
Device(config)# fips authorization-key key
```

Note that *key* is 128 bits, which is, 16 HEX byte key.

#### What to do next

After you enable FIPS, reboot the system to start operating in FIPS mode.

# Key Zeroization

A critical FIPS requirement is the capability to zeroize keys and passwords in the event of unsafe state triggers during FIPS mode of operation.

You can delete the FIPS authorization keys using the **no fips authorization-key** command in global configuration mode. This command deletes the key from flash. A reboot takes the system out of FIPS mode of operation.

If there is a security breach, use the **fips zeroize** command to delete all data including the running configuration, Trust Anchor Module, FIPS authorization keys, all ISE Server certificates, and IOS image in flash.

The system reboots after this command is executed.

**Caution** FIPS zeroization is a critical step where all data is lost. Use it with caution.

Session keys are zeroized by the protocols programmatically.

```
Device(config)#fips zeroize

**Critical Warning** - This command is irreversible
and will zeroize the FVPK by Deleting the IOS
image and config files, please use extreme
caution and confirm with Yes on each of three
```

```
iterations to complete. The system will reboot
after the command executes successfully
Proceed ?? (yes/[no]):
```

# Disable FIPS Mode

You can disable FIPS mode using the **no fips authorization-key** command.

The **no fips authorization-key** command deletes the authorization key from flash. Note that the authorization key is operational until you reload the switch.

To completely remove the authorization key and disable FIPS mode, reload the switch.

```
Device> enable
Device# config terminal
Device(config)# no fips authorization-key
Device(config)# end
```

# Verify FIPS Configuration

Use the **show fips status** command to display the FIPS configuration information.

Use the **show fips authorization-key** command to display the hashed FIPS key.

**Note** FIPS configuration information does not appear when you list the active configuration using the **show running-config** command or when you list the startup configuration using the **show startup-config** command.

The following are sample outputs of the **show** commands:

```
Device# show fips authorization-key

FIPS: Stored key (16) : 11111111111111111111111111111111
```

```
Device#show romvar

ROMMON variables:
PS1="switch: "
BOARDID="24666"
SWITCH_NUMBER="1"
TERMLINES="0"
MOTHERBOARD_ASSEMBLY_NUM="73-18506-02"
MOTHERBOARD_REVISION_NUM="04"
MODEL_REVISION_NUM="P2A"
POE1_ASSEMBLY_NUM="73-16123-03"
POE1_REVISION_NUM="A0"
POE1_SERIAL_NUM="FOC21335EF2"
POE2_ASSEMBLY_NUM="73-16123-03"
POE2_REVISION_NUM="A0"
POE2_SERIAL_NUM="FOC21335EF3"
IMAGE_UPGRADE="no"
MAC_ADDR="F8:7B:20:77:F7:80"
MODEL_NUM="C9300-48UN"
```

```
MOTHERBOARD_SERIAL_NUM="FOC21351BC3"
BAUD="9600"
SYSTEM_SERIAL_NUM="FCW2138L0AF"
USB_SERIAL_NUM="FOC213609Y5"
STKPWR_SERIAL_NUM="FOC21360HTS"
STKPWR_ASSEMBLY_NUM="73-11956-08"
STKPWR_REVISION_NUM="B0"
USB_ASSEMBLY_NUM="73-16167-02"
USB_REVISION_NUM="A0"
TAN_NUM="68-101202-01"
TAN_REVISION_NUMBER="23"
VERSION_ID="P2A"
CLEI_CODE_NUMBER="ABCDEFGHIJ"
ECI_CODE_NUMBER="123456"
TAG_ID="E20034120133FC00062B0965"
IP_SUBNET_MASK="255.255.0.0"
TEMPLATE="access"
TFTP_BLKSIZE="8192"
ENABLE_BREAK="yes"
TFTP_SERVER="10.8.0.6"
DEFAULT_GATEWAY="10.8.0.1"
IP_ADDRESS="10.8.3.33"
CRASHINFO="crashinfo:crashinfo_RP_00_00_20180420-020851-PDT"
CALL_HOME_DEBUG="0000000000000"
IP_ADDR="172.21.226.35/255.255.255.0"
DEFAULT_ROUTER="10.5.49.254"
RET_2_RTS=""
FIPS_KEY="5AC9BCA165E85D9FA3F2E5FC96AD98E8F943FBAB79B93E78"
MCP_STARTUP_TRACEFLAGS="00000000:00000000"
AUTOREBOOT_RESTORE="0"
MANUAL_BOOT="yes"
<output truncated>
Device#
```

# Additional References for Secure Operation in FIPS Mode

### Standards and RFCs

| Standards/RFCs | Title |
|---|---|
| FIPS 140-2 | Security Requirements for Cryptographic Modules |

### Technical Assistance

| Description | Link |
| --- | --- |
| The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.<br><br>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.<br><br>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. | http://www.cisco.com/cisco/web/support/index.html |