

System Security Configuration Guide for Cisco ASR 9000 Series Routers, IOS XR Release 26.1.x, 26.2.x, 26.3.x

First Published: 2026-02-28

Last Modified: 2026-09-11

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.

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)

© 2026 Cisco Systems, Inc. All rights reserved.



CONTENTS

PREFACE

Preface xix

Changes to This Document xix

Communications, Services, and Additional Information xix

CHAPTER 1

New and Changed Feature Information 1

System Security Features Added or Modified in IOS XR Release 26.x.x 1

CHAPTER 2

YANG Data Models for System Security Features 3

Using YANG Data Models 3

CHAPTER 3

Configuring AAA Services 5

Information About Configuring AAA Services 6

User, User Groups, and Task Groups 6

User Categories 6

User Groups 7

Task Groups 8

Admin Access for NETCONF and gRPC Sessions 9

User Profile Mapping from XR VM to System Admin VM 10

How to Allow Read Access to Administration Data for NETCONF and gRPC Clients 11

Cisco IOS XR Software Administrative Model 12

Administrative Access 12

AAA Database 13

Remote AAA Configuration 14

AAA Configuration 14

Authentication 15

Password Types 18

Type 8 and Type 9 Passwords 18
Type 10 Password 18
AAA Password Security for FIPS Compliance 19
AAA Password Security Policies 19
Minimum Password Length for First User Creation 21
Task-Based Authorization 22
Task IDs 22
General Usage Guidelines for Task IDs 22
Task IDs for TACACS+ and RADIUS Authenticated Users 23
Task Maps 24
Privilege Level Mapping 26
XML Schema for AAA Services 26
About RADIUS 27
Network Security Situations in Which RADIUS is Unsuitable 27
RADIUS Operation 28
RADIUS with TLS protection 29
Restrictions for RADIUS with TLS Protection 31
Supported ciphers 31
Configure RADIUS with TLS protection 32
Differentiated Services Code Point (DSCP) Marking support for TACACS packets 33
Hold-Down Timer for TACACS+ 34
How to Configure AAA Services 36
Prerequisites for Configuring AAA Services 36
Restrictions for Configuring AAA Services 36
Configuring Task Groups 36
Task Group Configuration 37
Configuring User Groups 38
Configure First User on Cisco Routers 40
Configuring Users 41
Password Masking For Type 7 Password Authentication 43
Configure Type 8 and Type 9 Passwords 44
Configure Type 10 Password 45
Backward Compatibility for Password Types 47
Configure AAA Password Policy 47

Display Username for Failed Authentication for Telnet Protocols 50
Enable Display of Username for Failed Authentication 50
Password Policy to Restrict Consecutive Characters 52
How to Restrict Consecutive Characters for User Passwords and Secrets 53
Deprecation of Type 7 password and Type 5 secret 56
Configuring Router to RADIUS Server Communication 61
Configuring RADIUS Dead-Server Detection 65
Configuring Per VRF AAA 66
New Vendor-Specific Attributes (VSAs) 67
Configuring a TACACS+ Server 69
Configuring RADIUS Server Groups 71
Configuring TACACS+ Server Groups 74
Configure Per VRF TACACS+ Server Groups 76
View TACACS+ information in Router 78
Configuring AAA Method Lists 81
Configuring Authentication Method Lists 81
Configuring Authorization Method Lists 84
Configuring Accounting Method Lists 88
Generating Interim Accounting Records 90
Applying Method Lists for Applications 92
Enabling AAA Authorization 92
Enabling Accounting Services 93
Configuring Login Parameters 94
How to Configure Hold-Down Timer for TACACS+ 95
Configuration Examples for Configuring AAA Services 98
Configuring AAA Services: Example 98
TACACS+ with TLS protection 100
How TACACS+ with TLS protection work 101
Guidelines for TACACS+ with TLS protection 101
Restrictions for TACACS+ with TLS protection 102
Configure TACACS+ with TLS protection 102
Command Accounting 103
Model-based AAA 104
Prerequisites for Model Based AAA 105

Initial Operation 105
NACM Configuration Management and Persistence 106
Overview of Configuring NACM 106
NACM Rules 106
Enabling NACM 110
Verify the NACM Configurations 111
Disabling NACM 112
Dynamic Retrieval of NETCONF Access Control Model Policies 113
Configure Dynamic NACM 114
Router Configuration 115
TACACS+ Server Configuration 117
LDAP Server Configuration 118
Configure an LDAP source interface 119
Dynamic NACM using LDAP over TLS Authentication 120
Command Authorization Using Local User Account 125
Configure Command Authorization Using Local User Account 127
Feature Behavior and Use Case Scenarios 128
Additional References 130

CHAPTER 4 **Implementing Certification Authority Interoperability** 133
Prerequisites for Implementing Certification Authority 134
Restrictions for Implementing Certification Authority 134
Information About Implementing Certification Authority 135
Supported Standards for Certification Authority Interoperability 135
Certification Authorities 136
Purpose of CAs 136
IPSec Without CAs 136
IPSec with CAs 137
IPSec with Multiple Trustpoint CAs 137
How IPSec Devices Use CA Certificates 137
CA Registration Authorities 138
How to Implement CA Interoperability 138
Configuring a Router Hostname and IP Domain Name 138
RSA key pairs 139

Generate an RSA key pair 140
Importing a Public Key to the Router 141
System logs and error messages 142
Declaring a Certification Authority and Configuring a Trusted Point 144
Authenticating the CA 146
Multi-Tier Certificate Authority for Trustpoint Authentication 147
How to Use Multi-Tier CA for Trustpoint Authentication 147
Requesting Your Own Certificates 149
Configuring Certificate Enrollment Using Cut-and-Paste 150
Configuration Examples for Implementing Certification Authority Interoperability 152
Configuring Certification Authority Interoperability: Example 152
Expiry Notification for PKI Certificate 154
Learn About the PKI Alert Notification 154
Enable PKI Traps 155
Regenerate the Certificate 155
Automatic renewal of PKI certificate 157
Pre-requisites 158
Configuration Guidelines 159
Configuration Example 159
Integrating Cisco IOS XR and Crosswork Trust Insights 159
How to Integrate Cisco IOS XR and Crosswork Trust Insights 160
Generate Key Pair 163
Generate System Trust Point for the Leaf and Root Certificate 164
Generate Root and Leaf Certificates 165
System Certificates Expiry 167
Collect Data Dossier 167
Procedure to Test Key Generation and Data-signing with Different Key Algorithm 171
Verify Authenticity of RPM Packages Using Fingerprint 172
Support for Ed25519 Public-Key Signature System 174
Generate Crypto Key for Ed25519 Signature Algorithm 175
Integrate Cisco IOS XR with Cisco Crosswork Trust Insights using Ed25519 175
Where to Go Next 176
Additional References 176

CHAPTER 5 **Implementing Keychain Management** 179

Prerequisites for Configuring Keychain Management 179

Restrictions for Implementing Keychain Management 179

Information About Implementing Keychain Management 179

Lifetime of Key 180

How to Implement Keychain Management 180

Configuring a Keychain 181

Configuring a Tolerance Specification to Accept Keys 182

Configuring a Key Identifier for the Keychain 183

Configuring the Text for the Key String 184

Determining the Valid Keys 185

Configuring the Keys to Generate Authentication Digest for the Outbound Application Traffic 187

Configuring the Cryptographic Algorithm 188

Configuration Examples for Implementing Keychain Management 190

Configuring Keychain Management: Example 190

Additional References 190

CHAPTER 6 **Configure MACSec** 193

Understanding MACsec Encryption 194

Advantages of Using MACsec Encryption 195

Types of MACsec Implementation 195

Configure the EAP-derived MACsec CAK length 196

MKA Authentication Process 197

Hardware Support for MACSec 198

MACSec Limitations for Cisco ASR 9901 Routers 200

MACsec PSK 201

PQC key exchange groups for EAP-TLS 201

Ensure compatibility and FIPS compliance for PQC EAP-TLS configuration 202

Supported PQC and classical key exchange groups 202

Configure PQC key exchange groups for EAP-TLS 203

Troubleshoot EAP-TLS key exchange group negotiation 204

Fallback PSK 205

WAN MACsec 206

WAN MACsec Use Cases 206
MACsec Encryption on Layer 3 Subinterface 208
EAPoL Ether-Type and Destination-Address 209
Configuring and Verifying MACSec Encryption 209
Creating a MACsec Key Chain 210
Creating a User-Defined MACsec Policy 213
MACsec SAK Rekey Interval 216
MACsec Policy Exceptions 216
Applying MACsec Configuration on an Interface 217
Configuring and Verifying MACsec Encryption on Physical Interfaces 219
Configuring and Verifying MACsec Encryption on VLAN Subinterfaces 220
Configure EAPoL Ether-Type 0x876F 226
Configure EAPoL Destination Address 227
Verifying MACsec Encryption on IOS XR 229
Verifying MACsec Encryption on ASR 9000 242
Configuring and Verifying MACsec Encryption as a Service 246
Configuring MACsec as a Service 248
Configuring MACsec Service for L2VPN Network 249
Configuring MACsec Service for L3VPN Network 250
Applying MACsec Service Configuration on an Interface 252
Verifying MACsec Encryption on IOS XR 253
Verifying MACsec Encryption on ASR 9000 267
Quantum safe key distribution options for MACsec 270
Session key service 271
Configure SKS 271
Create the SKS profile on the server and client 272
Export the McEliece public key 273
Copy the McEliece public key to the server 274
Import the client McEliece public key 274
Generate the encrypted seed for the SKS engine with the client public key 275
Copy the encrypted seed to the client 276
Decrypt and import the seed to the sks engine 276
Attach the SKS profile to the MACsec policy 277
Understanding SKIP 278

Configure Point-to-Point MACsec Link Encryption using SKIP 280
Global MACsec Shutdown 284
Configure MACsec Shutdown 285
Verify MACsec Shutdown 285
Syslog Messages for MACsec Shutdown 286
MACsec ISSU 286
Restrictions for MACsec ISSU 287
Options to Control MKA Protocol Suspension Initiation for ISSU 287
MACsec SNMP MIB (IEEE8021-SECY-MIB) 292

CHAPTER 7 **Implementing Type 6 Password Encryption** 295
Restrictions for Type 6 password encryption 295
How to Implement Type 6 Password Encryption 295
Enabling Type6 Feature and Creating a Primary Key (Type 6 Server) 295
Implementing Key Chain for BGP Sessions (Type 6 Client) 298
Creating a BGP Session (Type 6 Password Encryption Use Case) 299

CHAPTER 8 **Implementing Lawful Intercept** 301
Prerequisites for Implementing Lawful Intercept 302
Restrictions for Implementing Lawful Intercept 303
Information About Lawful Intercept Implementation 304
Interception Mode 304
Overlapping Taps 305
Provisioning for VoIP Calls 305
Call Interception 305
Provisioning for Data Sessions 306
Data Interception 306
Lawful Intercept Topology 306
Layer 2 Lawful Intercept 307
Lawful Intercept on MPLS Layer 3 307
Scale or Performance Improvement 308
Intercepting IPv4 and IPv6 Packets 308
Lawful Intercept Filters 309
Intercepting Packets Based on Flow ID (Applies to IPv6 only) 309

Intercepting VRF (6VPE) and 6PE Packets 310
Encapsulation Type Supported for Intercepted Packets 310
Per Tap Drop Counter Support 311
High Availability for Lawful Intercept 311
Preserving TAP and MD Tables during RP Fail Over 311
Replay Timer 312
Installing Lawful Intercept (LI) Package 312
Installing and Activating the LI Package 312
Deactivating the LI PIE 313
Upgrade and Downgrade Scenarios for the Lawful Intercept package 314
How to Configure SNMPv3 Access for Lawful Intercept 317
Disabling SNMP-based Lawful Intercept 317
Configuring the Inband Management Plane Protection Feature 317
Enabling the Mediation Device to Intercept VoIP and Data Sessions 318
Adding MD and TAP Objects 320
Configuration Example for Inband Management Plane Feature Enablement 322
Configuring the Inband Management Plane Protection Feature: Example 322
Increased Lawful Interception rates 322
Configure increased Lawful Interception rates 323
Additional References 324

CHAPTER 9 **Implementing Management Plane Protection** 327
Prerequisites for Implementing Management Plane Protection 327
Restrictions for Implementing Management Plane Protection 327
Information About Implementing Management Plane Protection 328
Inband Management Interface 328
Out-of-Band Management Interface 328
Peer-Filtering on Interfaces 329
Control Plane Protection Overview 329
Management Plane 329
Management Plane Protection Feature 329
Benefits of the Management Plane Protection Feature 330
How to Configure a Device for Management Plane Protection 330
Configuring a Device for Management Plane Protection for an Inband Interface 330

Configuring a Device for Management Plane Protection for an Out-of-band Interface 333
Configuration Examples for Implementing Management Plane Protection 335
Configuring Management Plane Protection: Example 336
Additional References 337

CHAPTER 10 **Traffic Protection for Third-Party Applications 339**
gRPC Protocol 339
Limitations for Traffic Protection for Third-Party Applications 340
Prerequisites for Traffic Protection for Third-Party Applications Over GRPC 340
Configuring Traffic Protection for Third-Party Applications 340
Troubleshooting Traffic Protection for Third-Party Applications 341

CHAPTER 11 **Configuring Software Authentication Manager 343**
Prerequisites for Configuring Software Authentication Manager 343
Information about Software Authentication Manager 343
How to set up a Prompt Interval for the Software Authentication Manager 344

CHAPTER 12 **Implementing Secure Shell 347**
Prerequisites for Implementing Secure Shell 348
SSH and SFTP in Baseline Cisco IOS XR Software Image 349
Information About Implementing Secure Shell 349
SSH Server 349
SSH Client 350
SFTP Feature Overview 351
RSA Based Host Authentication 352
RSA Based User Authentication 353
SSHv2 Client Keyboard-Interactive Authentication 353
Guidelines and Restrictions for Implementing Secure Shell 354
How to Implement Secure Shell 355
Configuring SSH 355
Automatic generation of SSH host-key pairs 359
Configure the Allowed SSH Host-Key Pair Algorithms 360
Ed25519 Public-Key Signature Algorithm Support for SSH 362
How to Generate Ed25519 Public Key for SSH 363

Configuring the SSH Client 363
Order of SSH Client Authentication Methods 365
How to Set the Order of Authentication Methods for SSH Clients 366
Configuring CBC Mode Ciphers 366
Configuration Examples for Implementing Secure Shell 367
Configuring Secure Shell: Example 367
Multi-channeling in SSH 368
Restrictions for Multi-channeling Over SSH 368
Client and Server Interaction Over Multichannel Connection 368
Configure Client for Multiplexing 369
SSH Configuration Option to Restrict Cipher Public Key and HMAC Algorithm 370
Disable HMAC Algorithm 371
Enable Cipher Public Key 371
SSH client strict host key checks 373
Restrictions for SSH client strict host key check 374
Configure SSH client strict host key checking 375
SSH client knownhosts commands 376
User Configurable Maximum Authentication Attempts for SSH 377
Configure Maximum Authentication Attempts for SSH 378
X.509v3 Certificate-based Authentication for SSH 379
Configure X.509v3 Certificate-based Authentication for SSH 382
OpenSSH Certificate based Authentication for Router 387
Feature Highlights 389
Prerequisites 389
Configuration Example 389
Public Key-Based Authentication of SSH Clients 397
Enable Public Key-Based Authentication of SSH Client 398
Public key-based Authentication to SSH Server on Routers 401
Guidelines and Restrictions for Public key-based authentication to Routers 402
Configure Public key-based Authentication to Routers 403
Delete Public Keys in the Routers 405
Multi-Factor Authentication for SSH 406
Multi-Factor Authentication Workflow 407
Set Up Multi-Factor Authentication for SSH 408

Configure Duo System for MFA 408
Configure Duo Authentication Proxy for MFA 409
Configure ISE for MFA 409
Configure RADIUS Server Attributes for MFA 410
Verify MFA Set-up for SSH Connection 410
Selective Authentication Methods for SSH Server 411
Disable SSH Server Authentication Methods 411
SSH Port Forwarding 412
How to Enable SSH Port Forwarding 414
Non-Default SSH Port 417
How to Configure Non-Default SSH Port 418
SSH server timeouts 421
Unused connection timeout for SSH sessions 421
Set unused connection timeout for SSH sessions 422
Channel timeout for SSH sessions 425
Set channel timeout for SSH sessions 426
Additional References 428

CHAPTER 13 **Layer 2 Security Features** 431
Security Features for Layer 2 VPLS Bridge Domains 431

CHAPTER 14 **Implementing Traffic Storm Control under a VPLS Bridge** 433
Prerequisites for Implementing Traffic Storm Control 433
Restrictions for Implementing Traffic Storm Control 434
Information About Implementing Traffic Storm Control 434
Understanding Traffic Storm Control 434
Traffic Storm Control Defaults 435
Supported Traffic Types for Traffic Storm Control 435
Supported Ports for Traffic Storm Control 435
Traffic Storm Control Thresholds 436
Traffic Storm Control Drop Counters 436
How to Configure Traffic Storm Control 436
Enabling Traffic Storm Control on an AC under a Bridge 436
Enabling Traffic Storm Control on a PW under a Bridge 438

Enabling Traffic Storm Control on a Bridge Domain 440
Clearing Traffic Storm Control Drop Counters 441
Configuration Examples for Traffic Storm Control 442
Configuring Traffic Storm Control on an AC: Example 442
Configuring Traffic Storm Control on an Access PW: Example 443
Configuring Traffic Storm Control on the Bridge Domain: Example 445
Additional References 445

CHAPTER 15 **Configuring FIPS Mode** 447
Prerequisites for Configuring FIPS 448
How to Configure FIPS 449
Enabling FIPS mode 449
Configuring FIPS-compliant Keys 450
Configuring FIPS-compliant Key Chain 451
Configuring FIPS-compliant Certificates 452
Configuring FIPS-compliant OSPFv3 453
Configuring FIPS-compliant SNMPv3 Server 455
Configuring FIPS-compliant SSH Client and Server 456
Configuration Examples for Configuring FIPS 457
Configuring FIPS: Example 457

CHAPTER 16 **Implementing Cisco ASR 9000 vDDoS Mitigation** 459
Cisco ASR 9000 vDDoS Mitigation Overview 459
Information about Implementing Cisco ASR 9000 vDDoS Mitigation 460
Prerequisites for Implementing Cisco ASR 9000 vDDoS Mitigation 460
Restrictions for Implementing Cisco ASR 9000 vDDoS Mitigation 460
Configuring Cisco ASR 9000 vDDoS Mitigation 460
Installing Cisco ASR 9000 vDDoS Software 460
Configuring Interfaces for TMS Mitigation 461
Uninstalling the TMS Virtual Service 463

CHAPTER 17 **Implementing Secure Logging** 465
System Logging over Transport Layer Security (TLS) 465
Restrictions for Syslogs over TLS 467

Configuring Syslogs over TLS 467
Security template framework for TLS applications 469
Restriction for security template 471
Configuration guidelines for security template framework 471
How security template framework works 472
Configure security template 474
TLS RFC 5289 compliance for security template 478

CHAPTER 18 **Gated shell access** 479
Key features of gated shell access 479
How gated shell access works 480
Restrict shell access using a Cisco-signed Consent Token 481
Unrestrict shell access using a Cisco-signed Consent Token 482
Restrict shell access using an owner Consent Token 483
Unrestrict shell access using an owner Consent Token 484
Terminate a pending Consent Token authorization request 485
Access a restricted shell 486

CHAPTER 19 **SSD Encryption** 487
SSD Encryption 487
DM-Crypt 488
AES-NI Support 488
CryptSetup 489
Encrypted Logical Volume 489
SSD Binding 490
Data Zeroization 490

CHAPTER 20 **Cisco MASA Service** 491
Why Do I Need Cisco MASA? 492
Use Cases for Ownership Vouchers 492
Authentication Flow 493
Interacting with the MASA Server 495
Interacting with MASA Through Web Application 497
Interacting with MASA Through REST APIs 500

Interaction with MASA through gRPC 501
Workflow to Provision a Router Using Ownership Voucher 502

CHAPTER 21 EST protocol for Automated Certificate Provisioning 505
EST protocol for automated certificate provisioning 505
Configure the EST protocol 510

CHAPTER 22 Upgrade CURL libraries for HTTP client 513
Key benefits for libcurl upgrades 513
Support and restrictions for libcurl upgrade 514
How libcurl upgrade works 514



Preface

This guide describes the configuration and examples for system security. For system security command descriptions, usage guidelines, task IDs, and examples, refer to the *System Security Command Reference for Cisco ASR 9000 Series Routers*.

From Release 6.1.2 onwards, Cisco introduces support for the 64-bit Linux-based IOS XR operating system. Extensive feature parity is maintained between the 32-bit and 64-bit environments. Unless explicitly marked otherwise, the contents of this document are applicable for both the environments. For more details on Cisco IOS XR 64 bit, refer to the Release Notes for Cisco ASR 9000 Series Routers, Release 6.1.2 document.

The preface contains the following sections:

- Changes to This Document, on page xix
- Communications, Services, and Additional Information, on page xix

Changes to This Document

This table lists the technical changes made to this document since it was first released.

Table 1: Changes to This Document

Date	Summary
September 2026	Republished for Release 26.3.1
June 2026	Republished for Release 26.2.1
February 2026	Initial release of this document

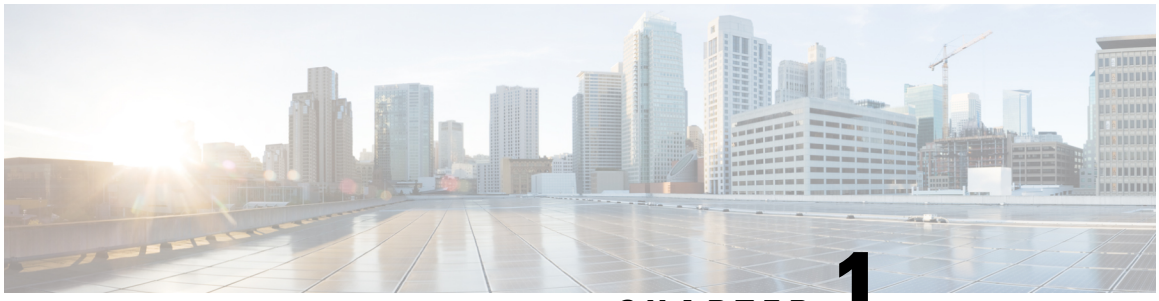
Communications, Services, and Additional Information

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business results you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class apps, products, solutions and services, visit Cisco Marketplace.

- To obtain general networking, training, and certification titles, visit Cisco Press.
- To find warranty information for a specific product or product family, access Cisco Warranty Finder.

Cisco Bug Search Tool

Cisco Bug Search Tool (BST) is a web-based tool that acts as a gateway to the Cisco bug tracking system that maintains a comprehensive list of defects and vulnerabilities in Cisco products and software. BST provides you with detailed defect information about your products and software.



CHAPTER 1

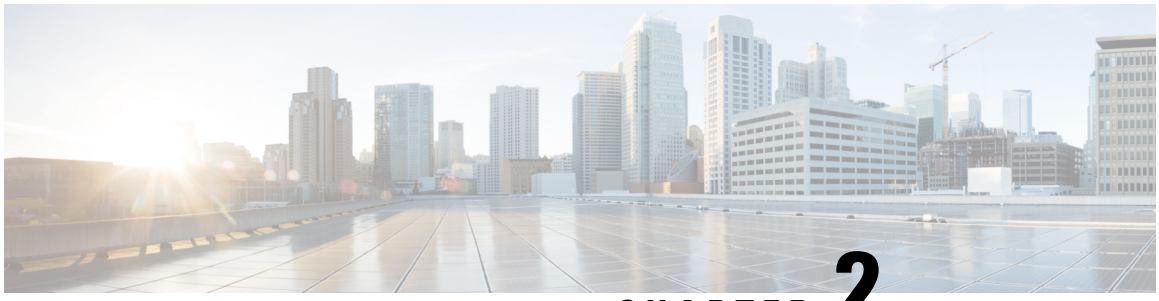
New and Changed Feature Information

This chapter lists all the features that have been added or modified in this guide. The table also contains references to these feature documentation sections.

- System Security Features Added or Modified in IOS XR Release 26.x.x, on page 1

System Security Features Added or Modified in IOS XR Release 26.x.x

Feature	Description	Changed in Release	Where Documented
Gated shell access	This feature was introduced.	Release 26.3.1	Gated shell access, on page 479
SSH client strict host key check	This feature was introduced.	Release 26.3.1	SSH client strict host key checks, on page 373
Upgrade CURL libraries for HTTP client	This feature was introduced.	Release 26.3.1	Upgrade CURL libraries for HTTP client, on page 513
PQC-enabled MACsec with EAP-TLS	This feature was introduced.	Release 26.3.1	PQC key exchange groups for EAP-TLS, on page 201
Syslog warnings for RSA keys and DSA keys	This feature was modified.	Release 26.1.1	RSA key pairs
SSH key strength: 3072 bits by default	This feature was introduced.	Release 26.1.1	Automatic generation of SSH host-key pairs , on page 359



CHAPTER 2

YANG Data Models for System Security Features

This chapter provides information about the YANG data models for System Security features.

• Using YANG Data Models, on page 3

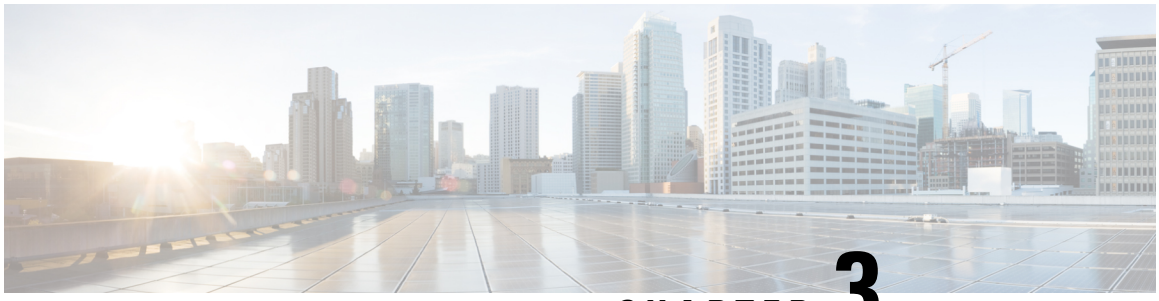
Using YANG Data Models

Cisco IOS XR supports a programmatic way of configuring and collecting operational data of a network device using YANG data models. Although configurations using CLIs are easier and human-readable, automating the configuration using model-driven programmability results in scalability.

The data models are available in the release image, and are also published in the Github repository. Navigate to the release folder of interest to view the list of supported data models and their definitions. Each data model defines a complete and cohesive model, or augments an existing data model with additional XPaths. To view a comprehensive list of the data models supported in a release, navigate to the **Available-Content.md** file in the repository.

You can also view the data model definitions using the YANG Data Models Navigator tool. This GUI-based and easy-to-use tool helps you explore the nuances of the data model and view the dependencies between various containers in the model. You can view the list of models supported across Cisco IOS XR releases and platforms, locate a specific model, view the containers and their respective lists, leaves, and leaf lists presented visually in a tree structure. This visual tree form helps you get insights into nodes that can help you automate your network.

To get started with using the data models, see the *Programmability Configuration Guide*.



CHAPTER 3

Configuring AAA Services

This module describes the implementation of the administrative model of *task-based authorization* used to control user access in the Cisco IOS XR software system. The major tasks required to implement task-based authorization involve configuring user groups and task groups.

User groups and task groups are configured through the Cisco IOS XR software command set used for authentication, authorization and accounting (AAA) services. Authentication commands are used to verify the identity of a user or principal. Authorization commands are used to verify that an authenticated user (or principal) is granted permission to perform a specific task. Accounting commands are used for logging of sessions and to create an audit trail by recording certain user- or system-generated actions.

AAA is part of the Cisco IOS XR software base package and is available by default.

Note For a complete description of the AAA commands listed in this module, see the *Authentication, Authorization, and Accounting Commands* module in *System Security Command Reference for Cisco ASR 9000 Series Routers*.

Feature History for Configuring AAA Services

Release	Modification
Release 3.7.2	This feature was introduced.
Release 4.1.0	Added the support for VRF aware TACACS+.
Release 6.3.1	Added the support for Type 8 and Type 9 passwords.
Release 6.3.2	Added the support for Command Accounting.
Release 7.0.1	Added the support for Type 10 password for 64-bit IOS XR.
Release 7.4.1	Added CLI commands to configure NACM rule-lists, rules and groups in addition to existing support for YANG data models.

- Information About Configuring AAA Services, on page 6
- How to Configure AAA Services, on page 36
- TACACS+ with TLS protection, on page 100
- Command Accounting, on page 103

- Model-based AAA, on page 104
- Overview of Configuring NACM, on page 106
- Disabling NACM, on page 112
- Dynamic Retrieval of NETCONF Access Control Model Policies, on page 113
- Dynamic NACM using LDAP over TLS Authentication, on page 120
- Command Authorization Using Local User Account, on page 125
- Additional References, on page 130

Information About Configuring AAA Services

This section lists all the conceptual information that a Cisco IOS XR software user must understand before configuring user groups and task groups through AAA or configuring Remote Authentication Dial-in User Service (RADIUS) or TACACS+ servers. Conceptual information also describes what AAA is and why it is important.

User, User Groups, and Task Groups

Cisco IOS XR software user attributes form the basis of the Cisco IOS XR software administrative model. Each router user is associated with the following attributes:

- User ID (ASCII string) that identifies the user uniquely across an administrative domain
- Length limitation of 253 characters for passwords and one-way encrypted secrets
- List of user groups (at least one) of which the user is a member (thereby enabling attributes such as task IDs). (See the Task IDs, on page 22 section)

User Categories

Router users are classified into the following categories:

- Root system user (complete administrative authority)
- Root Secure Domain Router (SDR) user (specific SDR administrative authority)
- SDR user (specific SDR user access)

Root System Users

The root system user is the entity authorized to “own” the entire router chassis. The root system user functions with the highest privileges over all router components and can monitor all secure domain routers in the system. At least one root system user account must be created during router setup. Multiple root system users can exist.

The root system user can perform any configuration or monitoring task, including the following:

- Configure secure domain routers.
- Create, delete, and modify root SDR users (after logging in to the secure domain router as the root system user). (See the Root SDR Users, on page 7 section.)

- Create, delete, and modify secure domain router users and set user task permissions (after logging in to the secure domain router as the root system user). (See the Secure Domain Router (SDR) Users, on page 7 section.)
- Access fabric racks or any router resource not allocated to a secure domain router, allowing the root system user to authenticate to any router node regardless of the secure domain router configurations.

Root SDR Users

A root SDR user controls the configuration and monitoring of a particular SDR. The root SDR user can create users and configure their privileges within the SDR. Multiple root SDR users can work independently. A single SDR may have more than one root SDR user.

A root SDR user can perform the following administrative tasks for a particular SDR:

- Create, delete, and modify secure domain router users and their privileges for the SDR. (See the Secure Domain Router (SDR) Users, on page 7 section.)
- Create, delete, and modify user groups to allow access to the SDR.
- Manage nearly all aspects of the SDR.

A root SDR user cannot deny access to a root system user. (See the Root System Users, on page 6 section.)

Secure Domain Router (SDR) Users

A SDR user has restricted access to an SDR as determined by the root-system user or root SDR user. The SDR user performs the day-to-day system and network management activities. The tasks that the secure domain router user is allowed to perform are determined by the task IDs associated with the user groups to which the SDR user belongs. (See the User Groups, on page 7 section.)

User Groups

A *user group* defines a collection of users that share a set of attributes, such as access privileges. Cisco IOS XR software allows the system administrator to configure groups of users and the job characteristics that are common in groups of users. Users are not assigned to groups by default hence the assignment needs to be done explicitly. A user can be assigned to more than one group.

Each user may be associated with one or more user groups. User groups have the following attributes:

- A user group consists of the list of task groups that define the authorization for the users. All tasks, except cisco-support, are permitted by default for root system users. (See the Root System Users, on page 6 section.)
- Each user task can be assigned read, write, execute, or debug permission.

Predefined User Groups

The Cisco IOS XR software provides a collection of user groups whose attributes are already defined. The predefined groups are as follows:

- **cisco-support:** This group is used by the Cisco support team.
- **maintenance:** Has the ability to display, configure and execute commands for network, files and user-related entities.
- **netadmin:** Has the ability to control and monitor all system and network parameters.

- **operator:** A demonstration group with basic privileges.
- **provisioning:** Has the ability to display and configure network, files and user-related entities.
- **read-only-tg:** Has the ability to perform any show command, but no configuration ability.
- **retrieve:** Has the ability to display network, files and user-related information.
- **root-lr:** Has the ability to control and monitor the specific secure domain router.
- **root-system:** Has the ability to control and monitor the entire system.
- **serviceadmin:** Service administration tasks, for example, Session Border Controller (SBC).
- **sysadmin:** Has the ability to control and monitor all system parameters but cannot configure network protocols.

The user group root-system has root system users as the only members. (See the Root System Users, on page 6 section.) The root-system user group has predefined authorization; that is, it has the complete responsibility for root-system user-managed resources and certain responsibilities in other SDRs.

To verify the individual permissions of a user group, assign the group to a user and execute the **show user tasks** command.

User-Defined User Groups

Administrators can configure their own user groups to meet particular needs.

User Group Inheritance

A user group can derive attributes from another user group. (Similarly, a task group can derive attributes from another task group). For example, when user group A inherits attributes from user group B, the new set of task attributes of the user group A is a union of A and B. The inheritance relationship among user groups is dynamic in the sense that if group A inherits attributes from group B, a change in group B affects group A, even if the group is not reinherited explicitly.

Task Groups

A task group is defined by a collection of task IDs. Task groups contain task ID lists for each class of action.

Each user group is associated with a set of task groups applicable to the users in that group. A user's task permissions are derived from the task groups associated with the user groups to which that user belongs.

Predefined Task Groups

The following predefined task groups are available for administrators to use, typically for initial configuration:

- **cisco-support:** Cisco support personnel tasks
- **netadmin:** Network administrator tasks
- **operator:** Operator day-to-day tasks (for demonstration purposes)
- **root-lr:** Secure domain router administrator tasks
- **root-system:** System-wide administrator tasks
- **sysadmin:** System administrator tasks

- **serviceadmin:** Service administration tasks, for example, SBC

User-Defined Task Groups

Users can configure their own task groups to meet particular needs.

Group Inheritance

Task groups support inheritance from other task groups. (Similarly, a user group can derive attributes from another user group. See the User Groups, on page 7 section.) For example, when task group A inherits task group B, the new set of attributes of task group A is the union of A and B.

Admin Access for NETCONF and gRPC Sessions

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
Admin Access for NETCONF and gRPC Sessions	Release 7.4.1	This feature allows all authorized users on XR VM to access the administration data on the router through NETCONF or gRPC interface, similar to accessing the CLI. This functionality works by internally mapping the task group of the user on XR VM to a predefined group on System Admin VM. Therefore, the NETCONF and gRPC users can access the admin-related information on the router even if their user profiles do not exist on System Admin VM. Prior to this release, only those users who were authorized on XR VM could access System Admin VM through CLI, by using the **admin** command. Users that were not configured on System Admin VM were denied access through the NETCONF or gRPC interfaces.

NETCONF is an XML-based protocol used over Secure Shell (SSH) transport to configure a network. Similarly, gRPC is an open-source remote procedure call framework. The client applications can use these protocols to request information from the router and make configuration changes to the router. Prior to Cisco IOS XR Software Release 7.4.1, users who use NETCONF, gRPC or any other configuration interface, other than CLI, to access the admin-related information on the router, had to belong to user groups that are configured on System Admin VM. Otherwise, the router would issue an UNAUTHORIZED access error message and deny access through that client interface.

By default, XR VM synchronizes only the first-configured user to System Admin VM. If you delete the first-user in XR VM, the system synchronizes the next user in the **root-lr** group (which is the highest privilege

group in XR VM for Cisco IOS XR 64-bit platforms) to System Admin VM only if there are no other users configured in System Admin VM. The system does not automatically synchronize the subsequent users to System Admin VM. Therefore, in earlier releases, users whose profiles did not exist in System Admin VM were not able to perform any NETCONF or gRPC operations on System Admin VM.

From Cisco IOS XR Software Release 7.4.1 and later, the system internally maps the users who are authorized on XR VM to System Admin VM of the router, based on the task table of the user on XR VM. With this feature, the NETCONF and gRPC users can access admin-related information on the router even if their user profiles do not exist on System Admin VM. By default, this feature is enabled.

To know more about NETCONF and gRPC operations, see the *Use NETCONF Protocol to Define Network Operations with Data Models* chapter and the *Use gRPC Protocol to Define Network Operations with Data Models* chapter in *Programmability Configuration Guide for Cisco ASR 9000 Series Routers*.

User Profile Mapping from XR VM to System Admin VM

User privileges to execute commands and access data elements on the router are usually specified using certain command rules and data rules that are created and applied on the user groups.

For details on user groups, command rules and data rules, see the *Create User Profiles and Assign Privilege* chapter in .

When the internal process for AAA starts or when you create the first user, the system creates the following set of predefined groups, command rules and data rules in System Admin VM. These configurations are prepopulated to allow users of different groups (such as **root-system**, **admin-r** and **aaa-r**) in System Admin VM.

You can use the **show running-configuration aaa** command to view the AAA configurations.

```
aaa authentication groups group aaa-r gid 100 users %%__system_user__%%
!
aaa authentication groups group admin-r gid 100 users %%__system_user__%%
!
aaa authentication groups group root-system gid 100 users "%%__system_user__%% "
!
aaa authorization cmdrules cmdrule 1 context * command * group root-system ops rx action
accept
!
aaa authorization cmdrules cmdrule 2 context * command "show running-config aaa" group aaa-r
 ops rx action accept
!
aaa authorization cmdrules cmdrule 3 context * command "show tech-support aaa" group aaa-r
 ops rx action accept
!
aaa authorization cmdrules cmdrule 4 context * command "show aaa" group   aaa-r ops rx
action accept
!
aaa authorization cmdrules cmdrule 5 context * command show group   admin-r ops rx action
accept
!
aaa authorization datarules datarule 1 namespace * context * keypath * group root-system
ops rwx action accept
!
aaa authorization datarules datarule 2 namespace * context * keypath /aaa group aaa-r ops
r action accept
!
aaa authorization datarules datarule 3 namespace * context * keypath /aaa group admin-r ops
 rwx action reject
!
aaa authorization datarules datarule 4 namespace * context * keypath / group admin-r ops r
```

```
 action accept
!
```

The administration CLI for the user works based on the above configurations. The **root-system** is the group with the highest privilege in System Admin VM. The **admin-r** group has only read and execute access to all data. The **aaa-r** group has access only to AAA data. With the introduction of the admin access feature for all users, NETCONF and gRPC applications can also access the administration data based on the above rules and groups.

User Profile Mapping Based on Task-ID

This table shows the internal mapping of XR VM users to System Admin VM. The users in XR VM belong to various user groups, such as **aaa**, **admin**, **root-lr**, and **root-system**.

XR VM User Group:Task-ID	System Admin VM User Group
aaa:rwxd	**aaa-r**
aaa:rwx	**aaa-r**
aaa:rw	**aaa-r**
aaa:wx	**aaa-r**
aaa:rx	**aaa-r**
aaa:r	**aaa-r**
aaa:w	**aaa-x**
aaa:x	**aaa-x**
root-system:rwxd	**root-system**
root-lr:rwxd	**root-system**
admin:rwxd	**admin-r**
admin:rwx	**admin-r**
admin:rw	**admin-r**
admin:r	**admin-r**

How to Allow Read Access to Administration Data for NETCONF and gRPC Clients

NETCONF and gRPC users access the administration data on the router through GET operations as defined by the respective protocols. To allow this read access to administration data for users belonging to **admin-r** group, you must configure a new command rule specifically for the NETCONF or gRPC client.

Configuration Example

```
Router#admin
sysadmin-vm:0_RP0#configure
sysadmin-vm:0_RP0(config)#aaa authorization cmdrules cmdrule 6
sysadmin-vm:0_RP0(config-cmdrule-6)#context netconf
sysadmin-vm:0_RP0(config-cmdrule-6)#command get
sysadmin-vm:0_RP0(config-cmdrule-6)#group admin-r
sysadmin-vm:0_RP0(config-cmdrule-6)#ops rx
```

```
sysadmin-vm:0_RP0(config-cmdrule-6)#action accept
sysadmin-vm:0_RP0(config)#commit
```

Running Configuration

```
aaa authorization cmdrules cmdrule 6
 context netconf
 command get
 group   admin-r
 ops     rx
 action  accept
!
```

Associated Command

- **aaa authorization (System Admin-VM)**

Cisco IOS XR Software Administrative Model

The router operates in two planes: the administration (admin) plane and secure domain router (SDR) plane. The admin (shared) plane consists of resources shared across all SDRs, while the SDR plane consists of those resources specific to the particular SDR.

The root-system user has the highest level of responsibility for the router. This user provisions secure domain routers and creates root SDR users. After being created, root SDR users take most of the responsibilities from the root-system user for the SDR. Root SDR users in turn can create secure domain router users. Root-system users and root SDR users have fixed permissions (task IDs) that cannot be changed by users.

Each SDR has its own AAA configuration including, local users, groups, and TACACS+ and RADIUS configurations. Users created in one SDR cannot access other SDRs unless those same users are configured in the other SDRs.

Administrative Access

Administrative access to the system can be lost if the following operations are not well understood and carefully planned. A lockout of all root-system users is a serious issue that requires a system reload to recover the password.

- Configuring authentication that uses remote AAA servers that are not available, particularly authentication for the console.

Note The **none** option without any other method list is not supported in Cisco IOS XR software.

- Removing the flash card from disk0:, or a disk corruption, may deny auxiliary port authentication, which can affect certain system debugging abilities. However, if the console is available, the system is still accessible.
- Configuring command authorization or EXEC mode authorization on the console should be done with extreme care, because TACACS+ servers may not be available or may deny every command, which locks the user out. This lockout can occur particularly if the authentication was done with a user not

known to the TACACS+ server, or if the TACACS+ user has most or all the commands denied for one reason or another.

To avoid a lockout, we recommend these:

- Before turning on TACACS+ command authorization or EXEC mode authorization on the console, make sure that the user who is configuring the authorization is logged in using the appropriate user permissions in the TACACS+ profile.
- If the security policy of the site permits it, use the **none** option for command authorization or EXEC mode authorization so that if the TACACS+ servers are not reachable, AAA rolls over to the **none** method, which permits the user to run the command.
- Make sure to allow local fallback when configuring AAA. See, Authorization Configuration, on page 84.
- If you prefer to commit the configuration on a trial basis for a specified time, you may do so by using the **commit confirmed** command, instead of direct **commit**.

AAA Database

The AAA database stores the users, groups, and task information that controls access to the system. The AAA database can be either local or remote. The database that is used for a specific situation depends on the AAA configuration.

Local Database

AAA data, such as users, user groups, and task groups, can be stored locally within a secure domain router. The data is stored in the in-memory database and persists in the configuration file. The stored passwords are encrypted.

Note The database is local to the specific secure domain router (SDR) in which it is stored, and the defined users or groups are not visible to other SDRs in the same system.

You can delete the last remaining user from the local database. If all users are deleted when the next user logs in, the setup dialog appears and prompts you for a new username and password.

Note The setup dialog appears only when the user logs into the console.

Remote Database

AAA data can be stored in an external security server, such as CiscoSecure ACS. Security data stored in the server can be used by any client (such as a network access server [NAS]) provided that the client knows the server IP address and shared secret.

Remote AAA Configuration

Products such as CiscoSecure ACS can be used to administer the shared or external AAA database. The router communicates with the remote AAA server using a standard IP-based security protocol (such as TACACS+ or RADIUS).

Client Configuration

The security server should be configured with the secret key shared with the router and the IP addresses of the clients.

User Groups

User groups that are created in an external server are not related to the user group concept that is used in the context of local AAA database configuration on the router. The management of external TACACS+ server or RADIUS server user groups is independent, and the router does not recognize the user group structure. The remote user or group profiles may contain attributes that specify the groups (defined on the router) to which a user or users belong, as well as individual task IDs. For more information, see the Task IDs for TACACS+ and RADIUS Authenticated Users, on page 23 section.

Configuration of user groups in external servers comes under the design of individual server products. See the appropriate server product documentation.

Task Groups

Task groups are defined by lists of permitted task IDs for each type of action (such as read, write, and so on). The task IDs are basically defined in the router system. Task ID definitions may have to be supported before task groups in external software can be configured.

Task IDs can also be configured in external TACACS+ or RADIUS servers.

AAA Configuration

This section provides information about AAA configuration.

Method Lists

AAA data may be stored in a variety of data sources. AAA configuration uses *method lists* to define an order of preference for the source of AAA data. AAA may define more than one method list and applications (such as login) can choose one of them. For example, console and auxiliary ports may use one method list and the vty ports may use another. If a method list is not specified, the application tries to use a default method list. If a default method list does not exist, AAA uses the local database as the source.

Rollover Mechanism

AAA can be configured to use a prioritized list of database options. If the system is unable to use a database, it automatically rolls over to the next database on the list. If the authentication, authorization, or accounting request is rejected by any database, the rollover does not occur and the request is rejected.

The following methods are available:

- Local: Use the locally configured database (not applicable for accounting and certain types of authorization)
- TACACS+: Use a TACACS+ server (such as CiscoSecure ACS)
- RADIUS: Use a RADIUS server

- Line: Use a line password and user group (applicable only for authentication)
- None: Allow the request (not applicable for authentication)

Note If the system rejects the authorization request and the user gets locked out, you can try to rollback the previous configuration or remove the problematic AAA configuration through auxiliary port. To log in to the auxiliary port, use the local username and password; not the tacacs+ server credentials. The **config_rollback -n 0x1** command can be used to rollback the previous configuration. If you are not able to access the auxiliary port, a router reload might be required in such scenarios.

Server Grouping

Instead of maintaining a single global list of servers, the user can form server groups for different AAA protocols (such as RADIUS and TACACS+) and associate them with AAA applications (such as PPP and EXEC mode).

Note In scenarios where multiple servers are within a TACACS AAA server group and multiple such groups exist within a remote database, the fallback behavior is when one server within a group is unreachable, and the system will default to the next server in that same group. However, when there is a mismatch in the shared secret between the router and a TACACS server, the router will not attempt to connect to the subsequent server in the group. Instead, it will bypass that group entirely and proceed to the next available method (group or local or none) based on the configuration.

Authentication

Authentication is the most important security process by which a principal (a user or an application) obtains access to the system. The principal is identified by a username (or user ID) that is unique across an administrative domain. The applications serving the user (such as or Management Agent) procure the username and the credentials from the user. AAA performs the authentication based on the username and credentials passed to it by the applications. The role of an authenticated user is determined by the group (or groups) to which the user belongs. (A user can be a member of one or more user groups.)

Authentication of Root System User

The root-system user can log in to any node in any secure domain router in the system. A user is a root-system user if he or she belongs to the root-system group. The root-system user may be defined in the local or remote AAA database.

Authentication of Non-Owner Secure Domain Router User

When logging in from a non-owner secure domain router, the root system user must add the "@admin" suffix to the username. Using the "@admin" suffix sends the authentication request to the owner secure domain router for verification. The owner secure domain router uses the methods in the list-name **remote** for choosing the authentication method. The **remote** method list is configured using the **aaa authentication login remote** *method1 method2...* command. (See the Configuring AAA Method Lists, on page 81 section.)

Authentication of Owner Secure Domain Router User

An owner secure domain router user can log in only to the nodes belonging to the specific secure domain router associated with that owner secure domain router user. If the user is member of a root-sdr group, the user is authenticated as an owner secure domain router user.

Authentication of Secure Domain Router User

Secure domain router user authentication is similar to owner secure domain router user authentication. If the user is not found to be a member of the designated owner secure domain router user group or root-system user group, the user is authenticated as a secure domain router user.

Authentication Flow of Control

AAA performs authentication according to the following process:

1. A user requests authentication by providing a username and password (or secret).
2. AAA verifies the user's password and rejects the user if the password does not match what is in the database.
3. AAA determines the role of the user (root system user, root SDR user, or SDR user).

   - If the user has been configured as a member of a root-system user group, then AAA authenticates the user as a root-system user.
   - If the user has been configured as a member of an owner secure domain router user group, then AAA authenticates the user as an owner secure domain router user.
   - If the user has not been configured as a member of a root-system user group or an owner secure domain router user group, AAA authenticates the user as a secure domain router user.

Clients can obtain a user's permitted task IDs during authentication. This information is obtained by forming a union of all task group definitions specified in the user groups to which the user belongs. Clients using such information typically create a session for the user (such as an API session) in which the task ID set remains static. Both the EXEC mode and external API clients can use this feature to optimize their operations. EXEC mode can avoid displaying the commands that are not applicable and an EMS application can, for example, disable graphical user interface (GUI) menus that are not applicable.

If the attributes of a user, such as user group membership and, consequently, task permissions, are modified, those modified attributes are not reflected in the user's current active session; they take effect in the user's next session.

Korn Shell Authentication

The korn shell (ksh) is the primary shell for the auxiliary port of the route processor (RP), standby RP, and distributed RP cards and for console and auxiliary ports of line cards (LCs) and service processors (SPs). The following are some of the characteristics of ksh authentication:

- For security reasons, ksh authentication allows only root-system users who have a secret configured. A root-system user with a normal password will not be authenticated because the normal password is two-way encrypted and poses a security risk because the password information is stored in the flash disk, which can be easily decrypted.
- Every time a root-system user with a secret is configured using the normal AAA CLI, that user is a valid ksh user and no separate configuration is required.

- Ksh does not authenticate TACACS+ or RADIUS users, even if they are root-system users.
- Ksh authentication uses a single user password database, which means when a root-system user on a dSC is configured using the normal AAA CLI, that user can log in using this username password in any card. This includes the RP, standby RP, LC, and SP.
- Ksh authentication cannot be turned off or bypassed after the card is booted. To bypass authentication, a user needs a reload of the card. (See the "Bypassing ksh Authentication" section for details).
- The ksh run from the console (using the **run** command) is not authenticated because the **run** command needs the root-system task ID. Because the user is already root-system, the user is not authenticated again.

Bypassing ksh Authentication

Although the authentication to ksh is lightweight and depends on very few processes, there are cases when ksh authentication needs to be bypassed, including the following:

- dSC (Active RP) disk0 corruption
- Loss of Qnet connectivity
- Inability to determine the node ID of the dSC (Active RP)

To bypass ksh authentication, the user has to set the ROMMON variable AUX_AUTHEN_LEVEL to 0 and then reload the image. A reboot is required only on the card that has to bypass authentication.

The ROMMON variable AUX_AUTHEN_LEVEL can have one of the following values:

- 0—Authentication will be bypassed on the card.
- 1—Loose authentication. Authentication is performed on a best-effort basis and permits the user to access ksh if the system cannot access authentication information successfully.
- 2—Strict authentication. This is the default state.

  Under no circumstances is authentication bypassed. Even if the authentication infrastructure is down, the system simply denies access.

For example, to bypass authentication on the card, enter the following:

```
rommon1> AUX_AUTHEN_LEVEL=0
rommon2> sync
rommon2> boot tftp:/ ...
```

Authentication Failure

In a system which is configured either with TACACS+ or RADIUS authentication with AAA configuration similar to the configuration below during the first login attempt or attempts, following a system reload, the login to the RP auxiliary port fails.

```
aaa authentication login default group tacacs+ group radius local
line template aux
login authentication default
```

This is because following the reload, the auxiliary port rejects login attempts with a valid TACACS+ configured *username* and *password*.

In such a scenario, the user has to first login with a valid locally configured *username* and *password*, and any login thereafter with TACACS+ configured *username* and *password*. Alternatively, if the user is connected to the auxiliary port via a terminal server, first clear the line used on the terminal server itself, and thereafter the user will be able to login to the auxiliary port with the TACACS+ configured *username* and *password*.

Password Types

In configuring a user and that user's group membership, you can specify two types of passwords: encrypted or clear text.

The router supports both two-way and one-way (secret) encrypted user passwords. Secret passwords are ideal for user login accounts because the original unencrypted password string cannot be deduced on the basis of the encrypted secret. Some applications (PPP, for example) require only two-way passwords because they must decrypt the stored password for their own function, such as sending the password in a packet. For a login user, both types of passwords may be configured, but a warning message is displayed if one type of password is configured while the other is already present.

If both secret and password are configured for a user, the secret takes precedence for all operations that do not require a password that can be decrypted, such as login. For applications such as PPP, the two-way encrypted password is used even if a secret is present.

Type 8 and Type 9 Passwords

This feature provides the options for Type 8 and Type 9 passwords in AAA security services. The Type 8 and Type 9 passwords provide more secure and robust support for saving passwords w.r.t each username. Thus, in scenarios where a lot of confidential data need to be maintained, these encryption methods ensure that the admin and other user passwords are strongly protected.

The implementation of Type 8 password uses SHA256 hashing algorithm, and the Type 9 password uses scrypt hashing algorithm.

Note The Type 8 and Type 9 passwords are supported on the IOS XR 64-bit operating system starting from Cisco IOS XR Software Release 7.0.1. Prior to this release, it was supported only on the 32-bit operating system.

Type 10 Password

The Cisco IOS XR 64 bit software introduces the support for Type 10 password that uses **SHA512** encryption algorithm. The **SHA512** encryption algorithm provides improved security to the user passwords compared to the older algorithms such as **MD5** and **SHA256**. With this feature, **SHA512** becomes the default encryption algorithm for the passwords in user name configuration, even for the first user creation scenario. Prior to the introduction of Type 10 password, **MD5** was used as the default algorithm.

To configure Type 10 password, see Configure Type 10 Password, on page 45.

Restrictions for Type 10 Password Usage

These restrictions apply to the usage of Type 10 password:

- Backward compatibility issues such as configuration loss, authentication failure, and so on, are expected when you downgrade to lower versions that still use **MD5** or **SHA256** encryption algorithms. Convert

the passwords to Type 10 before such downgrades to minimize the impact of such issues. For details, see Backward Compatibility for Password Types, on page 47.

- In a first user configuration scenario or when you reconfigure a user, the system syncs only the Type 5 and Type 10 passwords from XR VM to System Admin VM and Host VM. It doesn't sync the Type 8 and Type 9 passwords in such scenarios.

AAA Password Security for FIPS Compliance

Cisco IOS XR Software introduces advanced AAA password strengthening policy and security mechanism to store, retrieve and provide rules or policy to specify user passwords. This password policy is applicable only for local users, and not for remote users whose profile information are stored in a third party AAA server. This policy is not applicable to secrets of the user. If both secret and password are configured for a user, then secret takes precedence, and password security policy does not have any effect on authentication or change of password for such users. This AAA password security policy works as such for Cisco IOS XR platforms. Whereas, this feature is supported only on XR VM, for Cisco IOS XR 64 bit platforms.

High Availability for AAA Password Security Policy

The AAA password policy configurations and username configurations remain intact across RP failovers or process restarts in the system. The operational data such as, lifetime of the password and lockout time of the user are not stored on system database or disk. Hence, those are not restored across RP failovers or process restarts. Users start afresh on the active RP or on the new process. Hence, users who were locked out before RP failover or process restart are able to login immediately after the failover or restart.

To configure AAA password policy, see Configure AAA Password Policy, on page 47.

AAA Password Security Policies

AAA password security for FIPS compliance consists of these policies:

Password Composition Policy

Passwords can be composed by any combination of upper and lower case alphabets, numbers and special characters that include: "!", "@", "#", "$", "%", "^", "&", "*", "(", and ")". Security administrator can also set the types and number of required characters that comprise the password, thereby providing more flexibility for password composition rules. The minimum number of character change required between passwords is 4, by default. There is no restriction on the upper limit of the number of uppercase, lowercase, numeric and special characters.

Password Length Policy

The administrator can set the minimum and maximum length of the password. The minimum configurable length in password policy is 2, and the maximum length is 253.

Password Lifetime Policy

The administrator can configure a maximum lifetime for the password, the value of which can be specified in years, months, days, hours, minutes and seconds. The configured password never expires if this parameter is not configured. The configuration remains intact even after a system reload. But, the password creation time is updated to the new time whenever the system reboots. For example, if a password is configured with a life time of one month, and if the system reboots on 29th day, then the password is valid for one more month

after the system reboot. Once the configured lifetime expires, further action is taken based on the password expiry policy (see the section on Password Expiry Policy).

Password Expiry Policy

If the password credential of a user who is trying to login is already expired, then the following actions occur:

- User is prompted to set the new password after successfully entering the expired password.
- The new password is validated against the password security policy.
- If the new password matches the password security policy, then the AAA data base is updated and authentication is done with the new password.
- If the new password is not compliant with the password security policy, then the attempt is considered as an authentication failure and the user is prompted again to enter a new password. The max limit for such attempts is in the control of login clients and AAA does not have any restrictions for that.

As part of password expiry policy, if the life time is not yet configured for a user who has already logged in, and if the security administrator configures the life time for the same user, then the life time is set in the database. The system checks for password expiry on the subsequent authentication of the same user.

Password expiry is checked only during the authentication phase. If the password expires after the user is authenticated and logged in to the system, then no action is taken. The user is prompted to change the password only during the next authentication of the same user.

Debug logs and syslog are printed for the user password expiry only when the user attempts to login. This is a sample syslog in the case of password expiry:

```
RP/0/RSP1/CPU0:Jun 21 09:13:34.241 : locald_DSC[308]: %SECURITY-LOCALD-5-USER_PASSWD_EXPIRED
 :
Password for user 'user12' has expired.
```

Password Change Policy

Users cannot change passwords at will. A password change is triggered in these scenarios:

- When the security administrator needs to change the password
- When the user is trying to get authenticated using a profile and the password for the profile is expired
- When the security administrator modifies the password policy which is associated to the user, and does not immediately change the password according to the policy

You can use the **show configuration failed** command to display the error messages when the password entered does not comply with the password policy configurations.

When the security administrator changes the password security policy, and if the existing profile does not meet the password security policy rules, no action is taken if the user has already logged in to the system. In this scenario, the user is prompted to change the password when he tries to get authenticated using the profile which does not meet the password security rules.

When the user is changing the password, the lifetime of the new password remains same as that of the lifetime that was set by the security administrator for the old profile.

When password expires for non-interactive clients (such as dot1x), an appropriate error message is sent to the clients. Clients must contact the security administrator to renew the password in such scenarios.

Service Provision after Authentication

The basic AAA local authentication feature ensures that no service is performed before a user is authenticated.

User Re-authentication Policy

A user is re-authenticated when he changes the password. When a user changes his password on expiry, he is authenticated with the new password. In this case, the actual authentication happens based on the previous credential, and the new password is updated in the database.

User Authentication Lockout Policy

AAA provides a configuration option, **authen-max-attempts**, to restrict users who try to authenticate using invalid login credentials. This option sets the maximum number of permissible authentication failure attempts for a user. The user gets locked out when he exceeds this maximum limit, until the lockout timer (**lockout-time**) is expired. If the user attempts to login in spite of being locked out, the lockout expiry time keep advancing forward from the time login was last attempted.

This is a sample syslog when user is locked out:

```
RP/0/RSP1/CPU0:Jun 21 09:21:28.226 : locald_DSC[308]: %SECURITY-LOCALD-5-USER_PASSWD_LOCKED
 :
User 'user12’ is temporarily locked out for exceeding maximum unsuccessful logins.
```

This is a sample syslog when user is unlocked for authentication:

```
RP/0/RSP1/CPU0:Jun 21 09:14:24.633 : locald_DSC[308]: %SECURITY-LOCALD-5-USER_PASSWD_UNLOCKED
 :
User 'user12' is unlocked for authentications.
```

Password Policy Creation, Modification and Deletion

Security administrators having write permission for AAA tasks are allowed to create password policy. Modification is allowed at any point of time, even when the policy is associated to a user. Deletion of password policy is not allowed until the policy is un-configured from the user.

After the modification of password policy associated with a user, security administrator can decide if he wants to change passwords of associated users complying to the password policy. Based on this, there are two scenarios:

- If the administrator configures the password, then the user is not prompted to change the password on next login.
- If the administrator does not configure the password, then the user is prompted to change the password on next login.

In either of the above cases, at every password expiry interval, the user is prompted to change the password on next login.

Debug messages are printed when password policies are created, modified and deleted.

Minimum Password Length for First User Creation

To authenticate the user for the first time, Cisco router prompts you to create a username and password, in any of the following situations:

- When the Cisco Router is booted for the very first time.
- When the router is reloaded with no username configuration.
- When the already existing username configurations are deleted.

By default, the minimum length for passwords in a Cisco router is limited to two characters. Due to noise on the console, there is a possibility of the router being blocked out. Therefore, the minimum length for password has been increased to six characters for a first user created on the box, in each of the situations described above. This reduces the probability of the router being blocked out. It avoids the security risks that are caused due to very small password length. For all other users created after the first one, the default minimum length for password is still two characters.

For more information about how to configure a first user, see Configure First User on Cisco Routers, on page 40.

Task-Based Authorization

AAA employs “task permissions” for any control, configure, or monitor operation through CLI or API. The Cisco IOS software concept of privilege levels has been replaced in Cisco IOS XR software by a task-based authorization system.

Task IDs

The operational tasks that enable users to control, configure, and monitor Cisco IOS XR software are represented by task IDs. A task ID defines the permission to run an operation for a command. Users are associated with sets of task IDs that define the breadth of their authorized access to the router.

Task IDs are assigned to users through the following means:

Each user is associated with one or more user groups. Every user group is associated with one or more *task groups*; in turn, every task group is defined by a set of task IDs. Consequently, a user’s association with a particular user group links that user to a particular set of task IDs. A user that is associated with a task ID can execute any operation associated with that task ID.

General Usage Guidelines for Task IDs

Most router control, configuration, or monitoring operation (CLI or XML API) is associated with a particular set of task IDs. Typically, a given CLI command or API invocation is associated with at least one or more task IDs. Neither the **config** nor the **commit** commands require any specific task id permissions. The configuration and commit operations do not require specific task ID permissions. Aliases also don't require any task ID permissions. You cannnot perform a configuration replace unless root-lr permissions are assigned. If you want to deny getting into configuration mode you can use the TACACS+ command authorization to deny the config command. These associations are hard-coded within the router and may not be modified. Task IDs grant permission to perform certain tasks; task IDs do not deny permission to perform tasks. Task ID operations can be one, all, or a combination of classes that are listed in this table.

Table 3: Task ID Classes

Operation	Description
Read	Specifies a designation that permits only a read operation.
Write	Specifies a designation that permits a change operation and implicitly allows a read operation.

Operation	Description
Execute	Specifies a designation that permits an access operation; for example ping and Telnet.
Debug	Specifies a designation that permits a debug operation.

The system verifies that each CLI command and API invocation conforms with the task ID permission list for the user. If you are experiencing problems using a CLI command, contact your system administrator.

Multiple task ID operations separated by a slash (for example read/write) mean that both operations are applied to the specified task ID.

Multiple task ID operations separated by a comma (for example read/write, execute) mean that both operations are applied to the respective task IDs. For example, the **copy ipv4 access-list** command can have the read and write operations applied to the acl task ID, and the execute operation applied to the *filesystem* task ID.

If the task ID and operations columns have no value specified, the command is used without any previous association to a task ID and operation. In addition, users do not have to be associated to task IDs to use ROM monitor commands.

Users may need to be associated to additional task IDs to use a command if the command is used in a specific configuration submode. For example, to execute the **show redundancy** command, a user needs to be associated to the system (read) task ID and operations as shown in the following example:

```
RP/0/RSP0/CPU0:router# show redundancy
```

Whereas, in administration EXEC mode, a user needs to be associated to both admin and system (read) task IDs and operations, as shown in the following example:

```
RP/0/RSP0/CPU0:router# admin
RP/0/RSP0/CPU0:router(admin)# show redundancy
```

Task IDs for TACACS+ and RADIUS Authenticated Users

Cisco IOS XR software AAA provides the following means of assigning task permissions for users authenticated with the TACACS+ and RADIUS methods:

- Specify the text version of the task map directly in the configuration file of the external TACACS+ and RADIUS servers.

  See the "Task Maps, on page 24" section for more details.

- Specify the privilege level in the configuration file of the external TACACS+ and RADIUS servers.

  See the "Privilege Level Mapping, on page 26" section for more details.

- Create a local user with the same username as the user authenticating with the TACACS+ and RADIUS methods.

- Specify, by configuration, a default task group whose permissions are applied to any user authenticating with the TACACS+ and RADIUS methods.

Task Maps

For users who are authenticated using an external TACACS+ server and RADIUS server, Cisco IOS XR software AAA supports a method to define task IDs remotely.

Format of the Task String

The task string in the configuration file of the TACACS+ server consists of tokens delimited by a comma (,). Each token contains either a task ID name and its permissions or the user group to include for this particular user, as shown in the following example:

task = " *permissions* : *taskid name* , # *usergroup name* , ..."



Note Cisco IOS XR software allows you to specify task IDs as an attribute in the external RADIUS or TACACS+ server. If the server is also shared by non-Cisco IOS XR software systems, these attributes are marked as optional as indicated by the server documentation. For example, CiscoSecure ACS and the freeware TACACS+ server from Cisco require an asterisk (*) instead of an equal sign (=) before the attribute value for optional attributes. If you want to configure attributes as optional, refer to the TACACS+ server documentation.

For example, to give a user named user1 BGP read, write, and execute permissions and include user1 in a user group named operator, the username entry in the external server's TACACS+ configuration file would look similar to the following:

```
user = user1{
member = some-tac-server-group
opap = cleartext "lab"
service = exec {
task = "rwx:bgp,#operator"
}
}
```

The r,w,x, and d correspond to read, write, execute and debug, respectively, and the pound sign (#) indicates that a user group follows.



Note The optional keyword must be added in front of "task" to enable interoperability with systems based on Cisco IOS software.

If CiscoSecure ACS is used, perform the following procedure to specify the task ID and user groups:

SUMMARY STEPS

1. Enter your username and password.
2. Click the **Group Setup** button to display the **Group Setup** window.
3. From the Group drop-down list, select the group that you want to update.
4. Click the **Edit Settings** button.
5. Use the scroll arrow to locate the Shell (exec) check box.
6. Check the **Shell (exec)** check box to enable the custom attributes configuration.
7. Check the **Custom attributes** check box.
8. Enter the following task string without any blank spaces or quotation marks in the field:

9. Click the **Submit + Restart** button to restart the server.

DETAILED STEPS

Procedure

Step 1 Enter your username and password.

Step 2 Click the **Group Setup** button to display the **Group Setup** window.

Step 3 From the Group drop-down list, select the group that you want to update.

Step 4 Click the **Edit Settings** button.

Step 5 Use the scroll arrow to locate the Shell (exec) check box.

Step 6 Check the **Shell (exec)** check box to enable the custom attributes configuration.

Step 7 Check the **Custom attributes** check box.

Step 8 Enter the following task string without any blank spaces or quotation marks in the field:

Example:

```
task=rwx:bgp,#netadmin
```

Step 9 Click the **Submit + Restart** button to restart the server.

The following RADIUS Vendor-Specific Attribute (VSA) example shows that the user is part of the sysadmin predefined task group, can configure BGP, and can view the configuration for OSPF:

Example:

```
user Auth-Type := Local, User-Password == lab
        Service-Type = NAS-Prompt-User,
        Reply-Message = "Hello, %u",
        Login-Service = Telnet,
        Cisco-AVPair = "shell:tasks=#sysadmin,rwx:bgp,r:ospf"
```

After user1 successfully connects and logs in to the external TACACS+ server with username user1 and appropriate password, the **show user tasks** command can be used in EXEC mode to display all the tasks user1 can perform. For example:

Example:

```
Username:user1
Password:
RP/0/RSP0/CPU0:router# show user tasks

Task:      basic-services  :READ    WRITE    EXECUTEDEBUG
Task:                 bgp  :READ    WRITE    EXECUTE
Task:                 cdp  :READ
Task:                diag  :READ
Task:          ext-access  :READ             EXECUTE
Task:             logging  :READ
```

Alternatively, if a user named user2, who does not have a task string, logs in to the external server, the following information is displayed:

Example:

```
Username:user2
Password:
RP/0/RSP0/CPU0:router# show user tasks
No task ids available
```

Privilege Level Mapping

For compatibility with TACACS+ daemons that do not support the concept of task IDs, AAA supports a mapping between privilege levels defined for the user in the external TACACS+ server configuration file and local user groups. Following TACACS+ authentication, the task map of the user group that has been mapped from the privilege level returned from the external TACACS+ server is assigned to the user. For example, if a privilege level of 5 is returned from the external TACACS server, AAA attempts to get the task map of the local user group priv5. This mapping process is similar for other privilege levels from 1 to 13. For privilege level 15, the root-system user group is used; privilege level 14 maps to the user group owner-sdr.

For example, with the Cisco freeware tac plus server, the configuration file has to specify *priv_lvl* in its configuration file, as shown in the following example:

```
user = sampleuser1{
    member = bar
    service = exec-ext {
        priv_lvl = 5
    }
}
```

The number 5 in this example can be replaced with any privilege level that has to be assigned to the user *sampleuser.*

With the RADIUS server, task IDs are defined using the Cisco-AVPair, as shown in the following example:

```
user = sampleuser2{
    member = bar
    Cisco-AVPair = "shell:tasks=#root-system,#cisco-support"{
        Cisco-AVPair = "shell:priv-lvl=10"
    }
}
```

XML Schema for AAA Services

The extensible markup language (XML) interface uses requests and responses in XML document format to configure and monitor AAA. The AAA components publish the XML schema corresponding to the content and structure of the data used for configuration and monitoring. The XML tools and applications use the schema to communicate to the XML agent for performing the configuration.

The following schema are published by AAA:

- Authentication, Authorization and Accounting configuration
- User, user group, and task group configuration
- TACACS+ server and server group configuration
- RADIUS server and server group configuration

About RADIUS

RADIUS is a distributed client/server system that secures networks against unauthorized access. In the Cisco implementation, RADIUS clients run on Cisco routers and send authentication and accounting requests to a central RADIUS server that contains all user authentication and network service access information.

RADIUS is a fully open protocol, distributed in source code format, that can be modified to work with any security system currently available on the market.

Cisco supports RADIUS under its AAA security paradigm. RADIUS can be used with other AAA security protocols, such as TACACS+, Kerberos, and local username lookup.



Note RADIUS is supported on all Cisco platforms, but some RADIUS-supported features run only on specified platforms.

RADIUS has been implemented in a variety of network environments that require high levels of security while maintaining network access for remote users.

Use RADIUS in the following 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. In one case, RADIUS has been used with Enigma security cards to validate users and grant access to network resources.
- Networks already using RADIUS. You can add a Cisco router with RADIUS to the network. This might be the first step when you make a transition to a Terminal Access Controller Access Control System Plus (TACACS+) server.
- Networks in which a user must access only a single service. Using RADIUS, you can control user access to a single host, utility such as Telnet, or protocol such as Point-to-Point Protocol (PPP). For example, when a user logs in, RADIUS identifies this user as having authorization to run PPP using IP address 10.2.3.4 and the defined access list is started.
- Networks that require resource accounting. You can use RADIUS accounting independent of RADIUS authentication or authorization. The RADIUS accounting functions allow data to be sent at the start and end of services, indicating the amount of resources (such as time, packets, bytes, and so on) used during the session. An Internet service provider (ISP) might use a freeware-based version of RADIUS access control and accounting software to meet special security and billing needs.
- Networks that support preauthentication. Using the RADIUS server in your network, you can configure AAA preauthentication and set up the preauthentication profiles. Preauthentication enables service providers to better manage ports using their existing RADIUS solutions and to efficiently manage the use of shared resources to offer differing service-level agreements.

Network Security Situations in Which RADIUS is Unsuitable

RADIUS is not suitable in the following network security situations:

- Multiprotocol access environments. RADIUS does not support the following protocols:
  - AppleTalk Remote Access (ARA)
  - NetBIOS Frame Control Protocol (NBFCP)
  - NetWare Asynchronous Services Interface (NASI)
  - X.25 PAD connections
- Router-to-router situations. RADIUS does not provide two-way authentication. RADIUS can be used to authenticate from one router to a router other than a Cisco router if that router requires RADIUS authentication.
- Networks using a variety of services. RADIUS generally binds a user to one service model.

RADIUS Operation

When a user attempts to log in and authenticate to an access server using RADIUS, the following steps occur:

1. The user is prompted for and enters 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:
   a. ACCEPT—The user is authenticated.
   a. REJECT—The user is not authenticated and is prompted to reenter the username and password, or access is denied.
   a. CHALLENGE—A challenge is issued by the RADIUS server. The challenge collects additional data from the user.
   a. CHANGE PASSWORD—A request is issued by the RADIUS server, asking the user to select a new password.

   The ACCEPT or REJECT response is bundled with additional data used for EXEC mode or network authorization. You must first complete RADIUS authentication before using RADIUS authorization. The additional data included with the ACCEPT or REJECT packets consists of the following:

- Services that the user can access, including Telnet, rlogin, or local-area transport (LAT) connections, and PPP, Serial Line Internet Protocol (SLIP), or EXEC mode services.
- Connection parameters, including the host or client IP address, access list, and user timeouts.

RADIUS with TLS protection

Table 4: Feature History Table

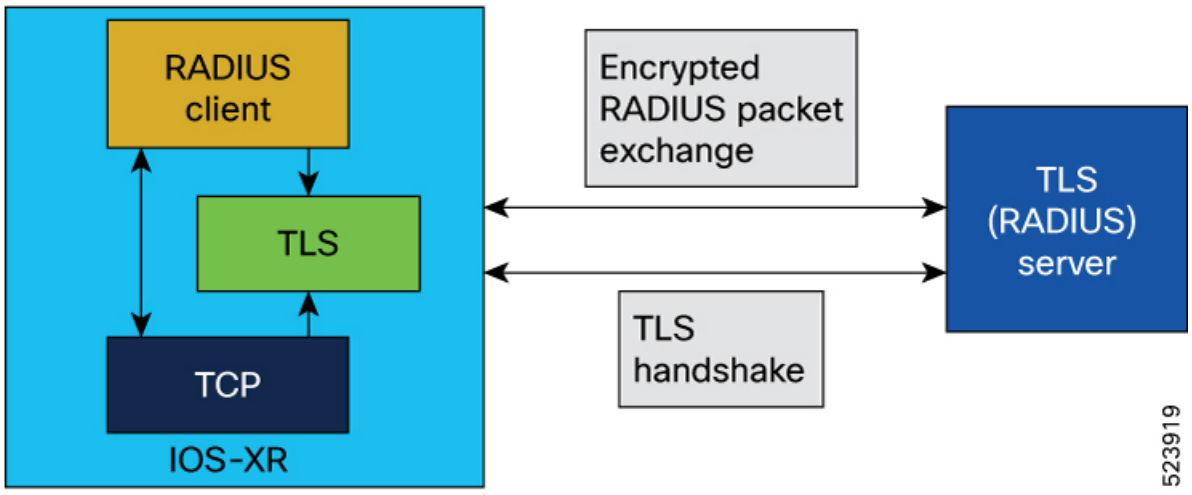
Feature Name	Release Information	Feature Description

RADIUS with TLS protection	Release 24.4.1	Remote Authentication Dial-In User Service (RADIUS) packets are now less vulnerable to security risks, including data exposure, replay attacks, weak authentication, and encryption weaknesses. This is because we have enabled support for RADIUS with TLS protection.
		
		You can configure the RADIUS protocol on the router to redirect RADIUS packets to a remote server over TLS for Authentication, Authorization, and Accounting (AAA) services.
		
		The feature introduces these changes:
		
		CLI:
		
		• The keyword **radsec-server** is introduced in the **radius-server host** command.
		
		YANG Data Models:
		
		• New Xpath for
		
		`Cisco-IOS-XR-um-aaa-cfg.yang`
		
		• New Xpath for
		
		`Cisco-IOS-XR-aaa-lib-cfg.yang`
		
		(see GitHub, YANG Data Models Navigator)

Topology for RADIUS with TLS protection

Traditionally, RADIUS has been used for Authentication, Authorization, and Accounting (AAA). However, to meet modern security demands it is important to enhance its encryption and authentication. To increase the resilience against threats and maintain a secure network environment, TLS is now utilized as the transport protocol for RADIUS.

This feature supports TLS version 1.3.



Let's understand how you can establish RADIUS communication with TLS.

Establish TLS session: The RADIUS client initiates the process to establish a secure TLS session with the RADIUS server, which acts as the TLS server. This session is built upon a TCP socket that the RADIUS client creates.

Store TLS context: Upon successful TLS connection, the TLS context is preserved within the TLS connection context. This, in turn, is stored within the RADIUS context for the specified server, ensuring a persistent secure environment for subsequent communications.

RADIUS packet handling: The format of the RADIUS packet remains consistent with that used in RADIUS over TCP. The application constructs a RADIUS packet using standard methods. Instead of sending it via a TCP socket, the packet is handed over to the TLS layer for secure encapsulation. TLS effectively becomes the transport layer for RADIUS, hence the designation "RADIUS/TLS."

Secure data transmission: The RADIUS packets are securely transmitted over the TLS layer with data encryption/decryption.

This approach ensures that the RADIUS communications are secure, especially in roaming environments where the packets may pass through various administrative domains and untrusted networks. TLS provides a robust security layer, addressing the vulnerabilities associated with traditional RADIUS over UDP.

Optimized RADIUS session control:

- To manage RADIUS sessions effectively, the RADIUS client employs Path MTU discovery before initiating traffic.
- The RADIUS client avoids using the same source socket for both RADIUS/UDP and RADIUS/TLS traffic to different servers.
- Once a TLS session is established, TLS heartbeats monitor connectivity with the server.
- Additionally, an application-layer watchdog algorithm checks server responsiveness. The client proactively closes idle sessions; sessions indicated as inactive by the TLS heartbeat, or those with only watchdog traffic for three timeouts.
- RADIUS sessions are terminated when RADIUS packets fail validation or contain invalid authenticators. When a session fails validation, the session is validated again using the next specified failover mechanism.

Restrictions for RADIUS with TLS Protection

The list provides the restrictions that apply to RADIUS with TLS protection:

- Broadband Network Gateway (BNG) applications are not supported.
- The default destination port for RADIUS over TLS is TCP 2083 for authentication and accounting. There is no support for custom ports.
- The maximum number of concurrent TLS sessions supported is 50.
- RADIUS over TLS supports IPv4 addresses only.
- A combination of TLS, UDP, and DTLS server types under one server group over RADIUS is not recommended.

Supported ciphers

TLS ciphers are encryption algorithms that secure RADIUS traffic. Here are some supported TLS ciphers:

- TLS_AES_256_CGM_SHA384
- TLS_CHACHA20_POLY1305_SHA256
- TLS_AES_128_CGM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CGM_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_CGM_SHA384
- TLS_DHE_RSA_WITH_AES_256_CGM_SHA384
- TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256
- TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
- TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA

- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_GCM_SHA384
- TLS_RSA_WITH_AES_128_GCM_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA256
- TLS_RSA_WITH_AES_128_CBC_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_EMPTY_RENEGOTIATION_INFO_SCSV

The cipher suite negotiated between the client and the server when both support TLS 1.3 is:

- TLS_AES_256_GCM_SHA384

Configure RADIUS with TLS protection

To configure RADIUS with TLS protection, use the command **radius-server host** with keyword **radsec-server**.

Before you begin

Before configuring RADIUS with TLS protection, complete these steps on the Cisco router. See **Implementing Certification Authority Interoperability** for more information.

1. Configure a trustpoint.
2. Import the CA certificate.
3. Enroll the trustpoint and generate a client certificate on CA.
4. Import the client certificate.

Procedure

Step 1 Enter the hostname or IP address of the RADIUS server.

```
Router(Config)#radius-server host 209.165.201.1 auth-port 2083 acct-port 2083 radsec-server
```

Step 2 Enter the name of the trusted point so that the router can verify certificates issued to peers.

```
Router(config-radius-host)trustpoint test
```

Your router need not enroll with the CA that issued the certificates to the peers.

Step 3 Commit the changes.

```
Router(config-radius-host)#commit
```

Step 4 Verify that TLS is enabled by using the **show radius** command.

```
Router#show radius
Thu Jun 20 11:43:40.863 UTC
```

```
Global dead time: 0 minute(s)
Number of Servers: 3

Server: 209.165.201.1/2083/2083  is UP
  Address family: IPv4
  Total Deadtime: 0s Last Deadtime: 0s
  Timeout: 5 sec, Retransmit limit: 3
  Quarantined: No
  Authentication:
    0 requests, 0 pending, 0 retransmits
    0 accepts, 0 rejects, 0 challenges
    0 timeouts, 0 bad responses, 0 bad authenticators
    0 unknown types, 0 dropped, 0 ms latest rtt
    Throttled: 0 transactions, 0 timeout, 0 failures
    Estimated Throttled Access Transactions: 0
    Maximum Throttled Access Transactions: 0

    Automated TEST Stats:
        0 requests, 0 timeouts, 0 response, 0 pending
  Server-type: TLS
  Accounting:
    0 requests, 0 pending, 0 retransmits
    0 responses, 0 timeouts, 0 bad responses
    0 bad authenticators, 0 unknown types, 0 dropped
    0 ms latest rtt
    Throttled: 0 transactions, 0 timeout, 0 failures
    Estimated Throttled Accounting Transactions: 0
    Maximum Throttled Accounting Transactions: 0

    Automated TEST Stats:
        0 requests, 0 timeouts, 0 response, 0 pending
```

Step 5 Verify the configuration settings by using the **show running-configuration** command.

```
Router#show running-configuration radius-server
Fri Jun 21 02:59:40.238 UTC
radius-server host 209.165.201.1 auth-port 2083 acct-port 2083
radsec-server trustpoint test
!
```

Differentiated Services Code Point (DSCP) Marking support for TACACS packets

Differentiated Services is a Quality of Service (QoS) architecture that manages the data traffic in a network by using the principle of traffic classification. In this model, the traffic is divided into classes and the data packets are forwarded to the corresponding classes. Based on the priority of the network traffic, the different classes are managed.

To classify traffic, Differentiated Services uses Differentiated Services Code Point (DSCP). It is a 6-bit field in the Type of Service (ToS) byte in the IP header. Based on the DSCP value, the user is able to classify the data traffic and forward packets to the next destination.

You can set the value of DSCP. For a single connection, set the DSCP value on the socket while connecting to the server. In this way, all the outgoing packets will have the same DSCP value in their IP headers. For multiple connections, the DSCP value is set on the available open sockets. Use the **tacacs-server ipv4** command to set the DSCP value.

Hold-Down Timer for TACACS+

Table 5: Feature History Table

Feature Name	Release Information	Feature Description
Hold-Down Timer for TACACS+	Release 7.4.1	TACACS+ servers provide AAA services to the user. When a TACACS+ server becomes unreachable, the router sends the client request to another server, leading to considerable delay in addressing requests. To prevent this delay, you can set a hold-down timer on the router. The timer gets triggered after the router marks the TACACS+ server as down. During this period, the router does not select the server that is down for processing any client requests. When the timer expires, the router starts using that TACACS+ server for client transactions. This feature improves latency in providing AAA services to the user by limiting the client requests from being sent to unresponsive servers. This feature introduces the **holddown-time** command.

The TACACS+ server is a AAA server with which the router communicates to provide authentication, authorization, and accounting services for users. When a TACACS+ server goes down, the router is not made aware. After sending a AAA request, the client waits for a response from the server for a configured timeout. If the router does not receive a response within that time frame, it sends the request to the next available server or discards the request if no other servers are available. A new request also needs to follow the same procedure in the same order of servers. The overall process results in sending multiple requests to servers that are down and therefore delays the client request from reaching an active server.

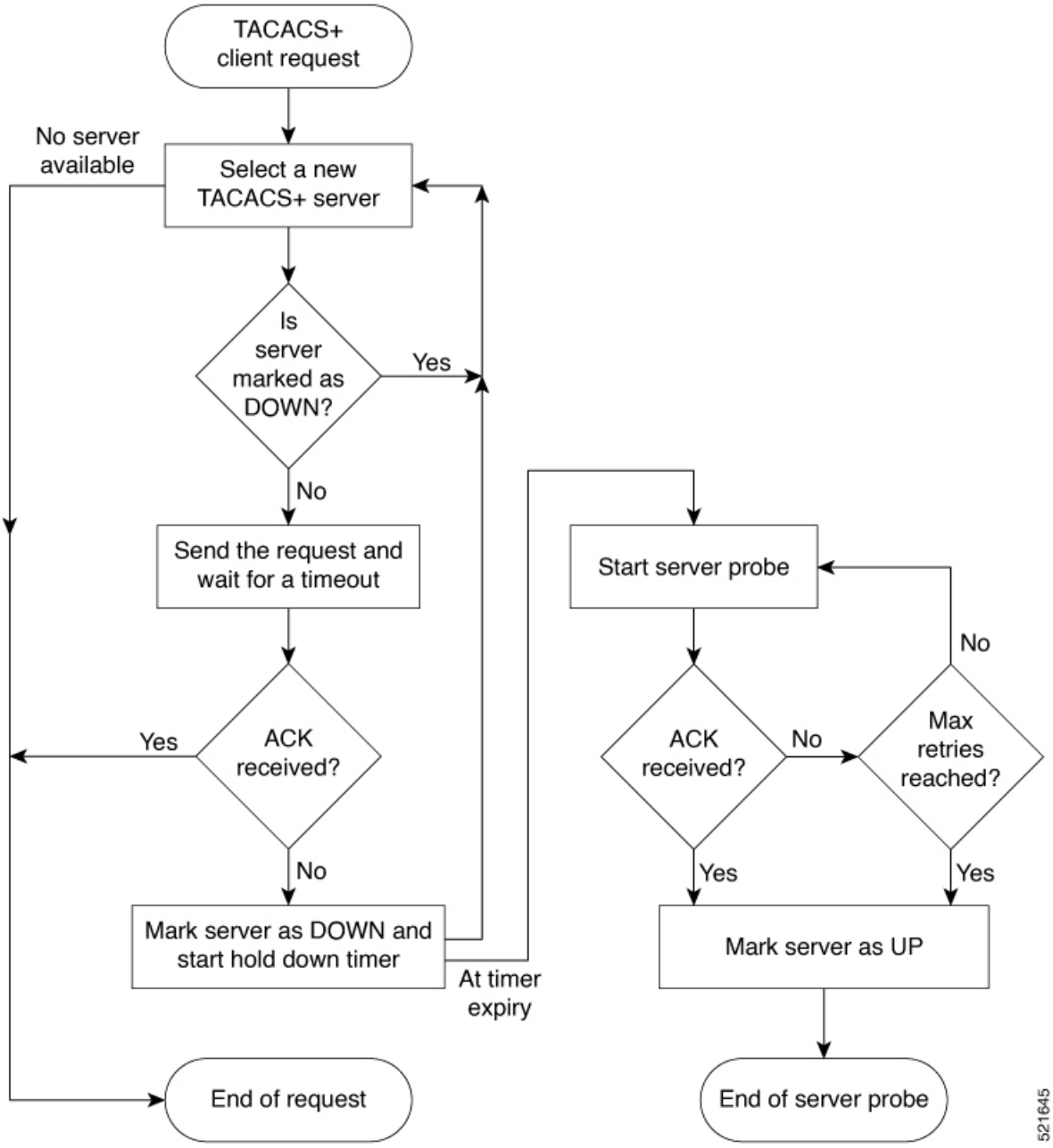
With the TACACS+ hold-down timer feature, you can mark an unresponsive TACACS+ server as down, and also set a duration for which the router does not use that server for further client transaction. After the timer expires, the router starts using that server again for processing client requests. This feature in turn allows you to control the participation of a TACACS+ server in AAA functions, without removing the TACACS+ server configuration from the router.

The hold-down timer value, in seconds, ranges from 0 to 1200. To enable hold-down timer, use the **holddown-time** command under the various configuration modes listed in the How to Configure Hold-Down Timer for TACACS+, on page 95 section.

How Does the Hold-Down Timer for TACACS+ Function?

The following image depicts the functionality of TACACS+ hold-down timer.

Figure 1: Work Flow of TACACS+ Hold-Down Timer



When a TACACS+ client request comes, the router selects a TACACS+ server and checks whether that server is marked as down. If the server is marked as down, the router selects another server until it finds an available server. If the server is not marked as down, the router sends the client request to that server. If the router does not receive an acknowledgment message from the server, it marks that server as down and initiates the hold-down timer. After the timer expires, an internal server probe begins, which checks the connectivity of the down server. The probe tries to connect to the server every 20 seconds, for a maximum of three times (these values are non-configurable). If connection is successful in any of these attempts, then the router marks that server as up, and ends the server probe. Even if the connection fails on all retries of the server probe, the

router still marks the server as up before exiting the server probe. After exiting the server probe, the router considers that server as available again to accept client requests.

If an unresponsive server is still not reachable after the hold-down timer continues, then the system continues to regard that server as being down, and does not use it for client transactions for some more time (that is, approximately, one minute). The router starts using that server again for further client transactions only after this short delay.

In case the TACACS+ server comes up while the hold-down timer is running, the router continues to consider that server as down until the timer expires.

How to Configure AAA Services

To configure AAA services, perform the tasks described in the following sections.

Prerequisites for Configuring AAA Services

The following are the prerequisites to configure AAA services:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- Establish a root system user using the initial setup dialog. The administrator may configure a few local users without any specific AAA configuration. The external security server becomes necessary when user accounts are shared among many routers within an administrative domain. A typical configuration would include the use of an external AAA security server and database with the local database option as a backup in case the external server becomes unreachable.

Restrictions for Configuring AAA Services

This section lists the restrictions for configuring AAA services.

Compatibility

Compatibility is verified with the Cisco freeware TACACS+ server and FreeRADIUS only.

Interoperability

Router administrators can use the same AAA server software and database (for example, CiscoSecure ACS) for the router and any other Cisco equipment that does not currently run Cisco IOS XR software. To support interoperability between the router and external TACACS+ servers that do not support task IDs, see the "Task IDs for TACACS+ and RADIUS Authenticated Users, on page 23" section.

Configuring Task Groups

Task-based authorization employs the concept of a *task ID* as its basic element. A task ID defines the permission to execute an operation for a given user. Each user is associated with a set of permitted router operation tasks identified by task IDs. Users are granted authority by being assigned to user groups that are in turn associated with task groups. Each task group is associated with one or more task IDs. The first configuration task in

setting up an authorization scheme to configure the task groups, followed by user groups, followed by individual users.

Task Group Configuration

Task groups are configured with a set of task IDs per action type.

Specific task IDs can be removed from a task group by specifying the **no** prefix for the **task** command.

The task group itself can be removed. Deleting a task group that is still referred to elsewhere results in an error.

Before you begin

Before creating task groups and associating them with task IDs, you should have some familiarity with the router list of task IDs and the purpose of each task ID. Use the **show aaa task supported** command to display a complete list of task IDs.

Note Only users with write permissions for the AAA task ID can configure task groups.

SUMMARY STEPS

1. **configure**
2. **taskgroup** *taskgroup-name*
3. **description** *string*
4. **task** {**read** | **write** | **execute** | **debug**} *taskid-name*
5. Repeat Step 4 for each task ID to be associated with the task group named in Step 2.
6. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**taskgroup** *taskgroup-name* **Example:** `RP/0/RSP0/CPU0:router(config)# taskgroup beta`	Creates a name for a particular task group and enters task group configuration submode. • Specific task groups can be removed from the system by specifying the **no** form of the **taskgroup** command.
Step 3	**description** *string* **Example:**	(Optional) Creates a description of the task group named in Step 2.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-tg)# description this is a sample task group description`	
Step 4	**task** {**read** | **write** | **execute** | **debug**} *taskid-name* **Example:** `RP/0/RSP0/CPU0:router(config-tg)# task read bgp`	Specifies a task ID to be associated with the task group named in Step 2. • Assigns **read** permission for any CLI or API invocations associated with that task ID and performed by a member of the task group. • Specific task IDs can be removed from a task group by specifying the **no** prefix for the **task** command.
Step 5	Repeat Step 4 for each task ID to be associated with the task group named in Step 2.	—
Step 6	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

After completing configuration of a full set of task groups, configure a full set of user groups as described in the Configuring User Groups section.

Configuring User Groups

User groups are configured with the command parameters for a set of users, such as task groups. Entering the **usergroup** command accesses the user group configuration submode. Users can remove specific user groups by using the **no** form of the **usergroup** command. Deleting a usergroup that is still referenced in the system results in a warning.

Before you begin



Note Only users associated with the WRITE:AAA task ID can configure user groups. User groups cannot inherit properties from predefined groups, such as root-system and owner-sdr.

SUMMARY STEPS

1. **configure**
2. **usergroup** *usergroup-name*
3. **description** *string*
4. **taskgroup** *taskgroup-name*
5. Repeat Step Step 4, on page 42 for each task group to be associated with the user group named in Step Step 2, on page 39.
6. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**usergroup** *usergroup-name* **Example:** `RP/0/RSP0/CPU0:router(config)# usergroup beta`	Creates a name for a particular user group and enters user group configuration submode. • Specific user groups can be removed from the system by specifying the **no** form of the **usergroup** command.
Step 3	**description** *string* **Example:** `RP/0/RSP0/CPU0:router(config-ug)# description this is a sample user group description`	(Optional) Creates a description of the user group named in Step Step 2, on page 39.
Step 4	**taskgroup** *taskgroup-name* **Example:** `RP/0/RSP0/CPU0:router(config-ug)# taskgroup beta`	Associates the user group named in Step Step 2, on page 42with the task group named in this step. • The user group takes on the configuration attributes (task ID list and permissions) already defined for the entered task group.
Step 5	Repeat Step Step 4, on page 42 for each task group to be associated with the user group named in Step Step 2, on page 39.	—
Step 6	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session.

	Command or Action	Purpose
		• **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

After completing configuration of a full set of user groups, configure individual users as described in the Configuring Users, on page 41 section.

Configure First User on Cisco Routers

When a Cisco Router is booted for the very first time, and a user logs in for the first time, a root-system username and password must be created. Configure the root-system username and password, as described in the following procedure:

Step 1. Establish a connection to the Console port.

This initiates communication with the router. When you have successfully connected to the router through the Console port, the router displays the prompt:

```
Enter root-system username
```

Step 2. Type the username for the root-system login and press **Enter**.

Sets the root-system username, which is used to log in to the router.

Step 3. Type the password for the root-system login and press **Enter**.

Creates an encrypted password for the root-system username. This password must be at least six characters in length. The router displays the prompt:

```
Enter secret
```

Step 4. Retype the password for the root-system login and press **Enter**.

Allows the router to verify that you have entered the same password both times. The router displays the prompt:

```
Enter secret again
```

Note If the passwords do not match, the router prompts you to repeat the process.

Step 5. Log in to the router.

Establishes your access rights for the router management session.

Note In case of Router reload, when there is no stored username and password, you must create a new username and password.

For more information on minimum password length, see .

Example

The following example shows the root-system username and password configuration for a new router, and it shows the initial login:

```
/* Administrative User Dialog */
Enter root-system username: cisco
Enter secret:
Enter secret again:

RP/0/0/CPU0:Jan 10 12:50:53.105 : exec[65652]: %MGBL-CONFIG-6-DB_COMMIT : 'Administration
configuration committed by system'.
Use 'show configuration commit changes 2000000009' to view the changes. Use the 'admin'
mode 'configure' command to modify this configuration.

/* User Access Verification */
Username: cisco
Password:
RP/0/0/CPU0:ios#
```

The secret line in the configuration command script shows that the password is encrypted. When you type the password during configuration and login, the password is hidden.

Configuring Users

Perform this task to configure a user.

Each user is id entified by a username that is unique across the administrative domain. Each user should be made a member of at least one user group. Deleting a user group may orphan the users associated with that group. The AAA server authenticates orphaned users but most commands are not authorized.

From Cisco IOS XR Software Release 24.3.1 and later, the router synchronizes up to 100 valid Linux-compatible users to the Linux infrastructure (/etc/passwd file), and up to 20 users to the standby route processor (RP) in a dual-RP router setup.

SUMMARY STEPS

1. **configure**
2. **username** *user-name*
3. Do one of the following:
   - **password** [**0** | **7**] *password*
   - **secret** [**0** | **5**] *secret*
   - **secret** [**0** | **5** | **8** | **9**] *secret*
   - **secret 0** [**enc-type**] *secret*
   - **secret 0 enc-type** {**5** | **8** | **9**}*secret*
4. **group** *group-name*
5. Repeat Step 4, on page 42 for each user group to be associated with the user specified in Step 2, on page 42.
6. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**username** *user-name* **Example:** `RP/0/RSP0/CPU0:router(config)# username user1`	RSP0 Creates a name for a new user (or identifies a current user) and enters username configuration submode. • The *user-name* argument can be only one word. Spaces and quotation marks are not allowed.

Step 3	Do one of the following:
	• **password** [**0** | **7**] *password*
	• **secret** [**0** | **5**] *secret*
	• **secret** [**0** | **5** | **8** | **9**] *secret*
	• **secret 0** [**enc-type**] *secret*
	• **secret 0 enc-type** {**5** | **8** | **9**}*secret*
	Example:
	`RP/0/RSP0/CPU0:router(config-un)# password 0 pwd1`
	or
	`RP/0/RSP0/CPU0:router(config-un)# secret 0 sec1`
		Specifies a password for the user named in Step 2, on page 42.
		• Use the **secret** command to create a secure login password for the user names specified in Step 2, on page 42.
		• Entering **0** following the **password** command specifies that an unencrypted (clear-text) password follows. Entering **7** following the **password** command specifies that an encrypted password follows.
		• Entering **0** following the **secret** command specifies that a secure unencrypted (clear-text) password follows. Entering **5** following the **secret** command specifies that a secure encrypted password follows.
		• Entering **8** following the **secret** command specifies that a SHA256 encrypted password follows. Entering **9** following the **secret** command specifies that a scrypt encrypted password follows.
		The **enc-type** keyword under the **secret 0** command allows the user to specify the algorithm that can be used to encrypt the clear text user password.
		• Type **0** is the default for the **password** and **secret** commands.

Step 4	**group** *group-name* **Example:** `RP/0/RSP0/CPU0:router(config-un)# group sysadmin`	Assigns the user named in Step 2, on page 42 to a user group that has already been defined through the **usergroup** command.

	Command or Action	Purpose
		• The user takes on all attributes of the user group, as defined by that user group's association to various task groups. • Each user must be assigned to at least one user group. A user may belong to multiple user groups.
Step 5	Repeat Step 4, on page 42 for each user group to be associated with the user specified in Step 2, on page 42.	—
Step 6	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

After completing configuration of a full set of users, configure router to use the RADIUS server communication or TACACS+ servers (See the Configuring Router to RADIUS Server Communication, on page 61 or Configuring a TACACS+ Server, on page 69 section.)

Password Masking For Type 7 Password Authentication

Table 6: Feature History Table

Feature Name	Release Information	Feature Description
Password Masking	Release 7.3.1	With this feature, when you key in a password or secret, it is not displayed on the screen. This enhances security. The feature is enabled by default. The following options are added to the **username** command: • **masked-password** • **masked-secret**

When you key in a password, to ensure that it is not displayed on the screen, use the **masked-password** option. Details:

Use the **username** command as shown below, and enter the password.

The following command contains the username us3, and 0 to specify a cleartext password.

```
Router(config)# username us3 masked-password 0

Enter password:
Re-enter password:

Router(config)#commit
```

View the encrypted password:

```
Router# show run aaa
..

username us3
 password 7 105A1D0D
```

Enable Type 7 password authentication and enter the encrypted password 105A1D0D. You can also use a password encrypted earlier.

```
Router(config)# username us3 masked-password 7

Enter password:
Re-enter password:


Router(config)#commit
```

If there is a password mismatch between the two entries, an error message is displayed.

Configure Type 8 and Type 9 Passwords

When configuring a password, user has the following two options:

- User can provide an already encrypted value, which is stored directly in the system without any further encryption.
- User can provide a cleartext password that is internally encrypted and stored in the system.

The Type 5, Type 8, and Type 9 encryption methods provide the above mentioned options for users to configure their passwords.

For more information about configuring Type 8 and Type 9 encryption methods, see Configuring Users, on page 41 section.

Configuration Example

Directly configuring a Type 8 encrypted password:

```
Router(config)# username demo8
Router(config-un)#secret 8 $8$dsYGNam3K1SIJO$7nv/35M/qr6t.dVc7UY9zrJDWRVqncHub1PE9UlMQFs
```

Configuring a clear-text password that is encrypted using Type 8 encryption method:

```
Router(config)# username demo8
Router(config-un)#secret 0 enc-type 8 PASSWORD
```

Directly configuring a Type 9 encrypted password:

```
Router(config)# username demo9
Router(config-un)# secret 9 $9$nhEmQVczB7dqsO$X.HsgL6x1il0RxkOSSvyQYwucySCt7qFm4v7pqCxkKM
```

Configuring a clear-text password that is encrypted using Type 9 encryption method:

```
Router(config)# username demo9
Router(config-un)#secret 0 enc-type 9 PASSWORD
```

Password Masking For Type 5, Type 8, Type 9 And Type 10 Password Authentication

When you key in a password, to ensure that it is not displayed on the screen, use the **masked-secret** option. Steps:

Use the **username** command as shown below, and enter the password.

The following command contains the username us6, 0 to specify a cleartext password, and the encryption type (5, 8, 9, or 10).

```
Router(config)# username us6 masked-secret 0 enc-type 8

Enter secret:
Re-enter secret:

Router(config)# commit
```

View the encrypted secret:

```
Router# show running-config aaa
..
username us6
 secret 8 $8$mlcSk/Ae5Qu/5k$RJdI3SQ8B4iP7rdxxQvVlJVeRHSubZzcgcaLYxjg36s
```

Enter the username, 8 to specify Type 8 secret authentication, and enter the Type 8 secret. You can also use a secret encrypted earlier.

```
Router(config)# username us6 masked-secret 8

Enter secret:
Re-enter secret:

Router(config)# commmit
```

If there is a password mismatch between the two entries, an error message is displayed.

Related Topics

- Type 8 and Type 9 Passwords, on page 18

Associated Commands

- secret
- username

Configure Type 10 Password

You can use these options to configure Type 10 password (that uses **SHA512** hashing algorithm) for the user:

Configuration Example

From Release 7.0.1 and later, Type 10 is applied by default for the passwords when you create a user with a clear-text password.

```
Router#configure
Router(config)#username user10 secret testpassword
Router(config-un)#commit
```

Also, a new parameter '10' is available for the **secret** option under the **username** command to configure explicitly the Type 10 passwords.

```
Router#configure
Router(config)#username root secret 10
$6$9UvJidvsTEqgkAPU$3CL1Ei/F.E4v/Hi.UaqLwX8UsSEr9ApG6c5pzhMJmZtgW4jObAQ7meAwyhu5VM/aRFJqe/jxZG17h6xPrvJWf1
Router(config-un)#commit
```

In scenarios where you have to enter the clear-text password, you can specify the encryption algorithm to be used by using the **enc-type** keyword and the clear-text password as follows:

```
Router#configure
Router(config)#username user10 secret 0 enc-type 10 testpassword
Router(config-un)#commit
```

The preceding configuration configures the user with the Type10 password.

In System Admin VM, you can specify the Type 10 encrypted password as follows:

```
Router#admin
sysadmin-vm:0_RP0# configure
sysadmin-vm:0_RP0(config)# aaa authentication users user user10 password testpassword
sysadmin-vm:0_RP0(config)# commit
Commit complete.
sysadmin-vm:0_RP0(config)# end
sysadmin-vm:0_RP0# exit
Router#
```

Running Configuration

```
Router#show running-configuration username user10
!
username user10
secret 10
$6$9UvJidvsTEqgkAPU$3CL1Ei/F.E4v/Hi.UaqLwX8UsSEr9ApG6c5pzhMJmZtgW4jObAQ7meAwyhu5VM/aRFJqe/jxZG17h6xPrvJWf1
!
```

In System Admin VM:

```
sysadmin-vm:0_RP0#show running-configuration aaa authentication users user user10
Tue Jan 14 07:32:44.363 UTC+00:00
aaa authentication users user user10
password
$6$MMvhlj1CzSd2nJfB$Bbzvxzriwx4iLFg75w4zj15YK3yeoq5UoRyc1evtSX0c4EuaMlqK.v7E3zbY1yKKxkN6rXpQuhMJOuyRHItDc1
!
sysadmin-vm:0_RP0#
```

Similarly, you can use the **admin show running-configuration aaa authentication users user user10** command in XR VM, to see the details of the password configured for the user.

Related Topics

- Type 10 Password, on page 18
- Backward Compatibility for Password Types, on page 47

Associated Commands

- **username**
- **secret**

Backward Compatibility for Password Types

When you downgrade from Cisco IOS XR Software Release 7.0.1 to lower versions, you might experience issues such as configuration loss, authentication failure, termination of downgrade process or XR VM being down. These issues occur because Type 5 (MD5) is the default encryption for older releases.

It is recommended to follow these steps to avoid such backward compatibility issues during downgrade:

- Perform all install operations for the downgrade except the **install activate** step.
- Before performing the **install activate** step, take the backup of user configurations on both the VMs. You can use the **show running-configuration username | file harddisk:***/filename* command for the same.
- Delete all users on both the VMs and initiate the **install activate** step.
- When the router boots up with the lower version, it prompts for the first root-system user creation.
- After your login with the credentials of the first user, apply the previously saved configuration to both the VMs.

For example, consider an authentication failure scenario after a downgrade. The downgrade process does not affect any existing user name configuration with Type 5 secret. Such users can log in without any issue using the clear-text password. But, the users with Type 10 configuration might experience authentication failure, and may not be able to log in. In such cases, the system treats the whole string "10<space><sha512-hashed-text>" as a clear-text password and encrypts it to Type 5 (MD5) password. Use that "10<space><sha512-hashed-text> string as the password for that Type 10 user to log in. After you log in with the preceding step, you must explicitly configure the clear-text password in XR VM and System Admin VM as described in the Configuration Example section.

Configure AAA Password Policy

To configure the AAA password policy, use the **aaa password-policy** command in the global configuration mode.

Configuration Example

This example shows how to configure a AAA password security policy, *test-policy*. This *test-policy* is applied to a user by using the **username** command along with **password-policy** option.

```
RP/0/RSP0/CPU0:router(config)#aaa password-policy test-policy
RP/0/RSP0/CPU0:router(config-aaa)#min-length 8
RP/0/RSP0/CPU0:router(config-aaa)#max-length 15
RP/0/RSP0/CPU0:router(config-aaa)#lifetime months 3
RP/0/RSP0/CPU0:router(config-aaa)#min-char-change 5
RP/0/RSP0/CPU0:router(config-aaa)#authen-max-attempts 3
RP/0/RSP0/CPU0:router(config-aaa)#lockout-time days 1
RP/0/RSP0/CPU0:router(config-aaa)#commit

RP/0/RSP0/CPU0:router(config)#username user1 password-policy test-policy password 0 pwd1
```

Running Configuration

```
aaa password-policy test-policy
 min-length 8
 max-length 15
 lifetime months 3
 min-char-change 5
 authen-max-attempts 3
 lockout-time days 1
 !
```

Verification

Use this command to get details of the AAA password policy configured in the router:

```
RP/0/RSP0/CPU0:router#show aaa password-policy

Fri Feb  3 16:50:58.086 EDT
Password Policy Name : test-policy
  Number of Users : 1
  Minimum Length : 8
  Maximum Length : 15
  Special Character Len : 0
  Uppercase Character Len : 0
  Lowercase Character Len : 1
  Numeric Character Len : 0
  Policy Life Time :
    seconds : 0
    minutes : 0
    hours : 0
    days : 0
    months : 3
    years : 0
  Lockout Time :
    seconds : 0
    minutes : 0
    hours : 0
    days : 1
    months : 0
    years : 0
  Character Change Len : 5
  Maximum Failure Attempts : 3
```

Password Masking For AAA Password Policies

When you key in a password, to ensure that it is not displayed on the screen, use the **masked-password** option. Steps:

Create a AAA password security policy and enter the cleartext password.

In this example, a policy called *security* is created, and 0 is specified for a cleartext password.

```
Router(config)# aaa password-policy security
Router(config)# username us6 password-policy security masked-password 0

Enter password:
Re-enter password:

Router(config)#commit
```

View the encrypted password:

```
Router# show run aaa
..

aaa password-policy security
..
username us6
 password-policy security password 7 0835585A
```

Enter the username, 7 to specify Type 7 password authentication, and enter the password 0835585A. You can also use a password encrypted earlier.

```
Router(config)# username us6 password-policy test-policy masked-password 7

Enter password:
Re-enter password:


Router(config)#commit
```

If there is a password mismatch between the two entries, an error message is displayed.

Related Topic

- AAA Password Security for FIPS Compliance, on page 19

Associated Commands

- **aaa password-policy**
- **show aaa password-policy**
- **username**

Display Username for Failed Authentication for Telnet Protocols

Table 7: Feature History Table

Feature Name	Release Information	Feature Description
Display Username for Failed Authentication for Telnet Protocols	Release 7.10.1	With this feature, we have enhanced the security of the routers and introduced better tracking functionality to the router. The failed authentication sys log now displays the details of users who tried to log in but failed due to authentication failure. With this feature provisioned, the router can now display the user ID of both SSH and Telnet protocols. In earlier releases, this feature was available only for SSH protocols. This feature introduces the following change: **CLI: aaa display-login-failed-users**. **YANG DATA Model**: New XPaths for `Cisco-IOS-XR-um-aaa-task-user-cfg` (see Github, YANG Data Models Navigator)

Effective Cisco IOS XR Software Release 7.10.1, you can track the username of the users who tried to login to the router and their authentication failed in the failed authentication system logs. Prior to this release, this feature was available for SSH clients only. Now, this functionality is available for both SSH and Telnet clients. By default, the feature is disabled. When this feature is disabled, failed authentication sys logs displays the username as **unknown** for both SSH and Telnet. Once the feature is enabled, the failed authentication sys logs display the username of the users who tried to login to the router, and the login attempt was unsuccessful due to failed authentication.

Use the **aaa display-login-failed-users** command in Global Configuration mode to enable this feature.

Enable Display of Username for Failed Authentication

Configuration Example

```
Router#conf
Router(config)#aaa display-login-failed-users
Router(config)#commit
```

Running configuration

```
Router#show run aaa display-login-failed-users
!
aaa display-login-failed-users
!
```

Verification

This section shows example from sys logs where the user name is displayed for failed authentication after the configuration of this feature.

System logs for Telnet client:

```
RP/0/RP0/CPU0:Jul 18 14:36:39.789 UTC: exec[65957]:
%SECURITY-LOGIN-4-AUTHEN_FAILED : Failed authentication attempt by
user lab from 'console' on 'con0_RP0_CPU0'
```

System logs for SSH client:

```
RP/0/RP0/CPU0:Jul 18 14:38:17.596 UTC: SSHD_[66072]:
 %SECURITY-SSHD-4-INFO_FAILURE : Failed authentication attempt by
 user lab from '192.168.122.1' on 'vty0
```

Password Policy to Restrict Consecutive Characters

Table 8: Feature History Table

Feature Name	Release Information	Feature Description
Password Policy to Restrict Consecutive Characters	Release 7.7.1	We have enhanced the router security by enforcing a strong password policy for all users configured on the router. You can now specify a new password policy for the user that restricts the usage of a specific number of consecutive characters for the login passwords. These characters include English alphabets, the sequence of QWERTY keyboard layout, and numbers, such as, 'abcd', 'qwer', '1234', and so on. Apart from *passwords*, the feature is also applicable for *secrets*–the one-way encrypted secure login passwords that are not easy to decrypt to retrieve the original unencrypted password text. The password policy is applicable only for the users configured on the local AAA server on the router; not those configured on the remote AAA server. The feature introduces the **restrict-consecutive-characters** command.

Most often you create passwords and secrets which are easy to remember, such as the ones that use consecutive characters from English alphabets, or numbers. Such passwords and secrets are easy to compromise, thereby making the router vulnerable to security attacks. From Cisco IOS XR Software Release 7.7.1 and later, you can enhance the security of your user passwords and secrets by defining a password policy that restricts the usage of consecutive characters from English alphabets, QWERTY layout keyboard English alphabets, and numbers (such as, 'abcd', 'qwer', 'zyxw', '1234', and so on). You can also restrict a cyclic wrapping of the alphabet and the number (such as, 'yzab', 'opqw', '9012', and so on). The feature also gives you the flexibility to specify the number of consecutive alphabets or numbers to be restricted.

Certain key aspects of this feature are:

- The feature is disabled, by default.
- The security administrator must have *write* permission for AAA tasks to create the password policies.
- All password policies are applicable only to locally-configured users; not to users who are configured on remote AAA servers.

This table depicts the examples of valid and invalid passwords and secrets when the password policy to restrict consecutive characters (say, 4 in this example) is in place.

Use Case	Examples of Invalid Password or Secret	Examples of Valid Password or Secret
4 consecutive English alphabets	AbcD, ABCD, TestPQRS, DcbA, TestZYxW123, DCBA, ihgf	AbcPqR, Xyzdef, Yzab, zabC
4 consecutive English alphabets and decimal numbers from QWERTY keyboard layout	Qwer, QWER, Mnbv, aQwerm, Test1234, TestT7890, 5678, fghj	Opas, xzLk, sapo, saqw3210, Test9012
Restrict 4 consecutive English alphabets along with cyclic wrapping	Yzab, TestYZAB, zabc	1234, Qwer, QWER, Mnbv, aQwerm, Test1234, TestT0987
Restrict 4 consecutive English alphabets and numbers from QWERTY keyboard layout along with cyclic wrapping	9012, 8901, Test3210, TestT0987, Opqw, klas, dsal, Cxzm, nmzx	AbcD, ABCD, Yzab, TestYZAB, zabc

How to Restrict Consecutive Characters for User Passwords and Secrets

To enable the feature to restrict consecutive characters for user passwords and secrets, use the **restrict-consecutive-characters** command in *aaa password policy* configuration mode. To disable the feature, use the **no** form of the command.

You can use the optional keyword, **cyclic-wrap**, to restrict the cyclic wrapping of characters and numbers.

After creating the password policies, you must explicitly apply those policies to the user profiles so that the password policies take effect in the password and secret configuration.

Configuration Example

Enabling the feature using CLI:

```
Router(config)#aaa password-policy test-policy
Router(config-pp)#restrict-consecutive-characters english-alphabet 4
Router(config-pp)#restrict-consecutive-characters qwerty-keyboard 5
```

The keyword, **cyclic-wrap**, to restrict cyclic wrapping is an optional parameter. If configured, then the feature also restricts the cyclic wrapping of characters and numbers.

```
Router(config-pp)#restrict-consecutive-characters english-alphabet 4 cyclic-wrap
Router(config-pp)#restrict-consecutive-characters qwerty-keyboard 5 cyclic-wrap
```

Applying the password policy to the user profile:

```
Router(config)#username user1
Router(config-un)#policy test-policy
Router(config-un)#commit
```

Running Configuration

This is a sample running configuration that shows that you have configured a AAA password policy that restricts six consecutive characters from the QWERTY keyboard, and cyclic wrapping of four consecutive English alphabets.

```
Router(config-pp)#show running-config aaa password-policy
Tue May 17 10:53:16.532 UTC

!
aaa password-policy test-policy
 restrict-consecutive-characters qwerty-keyboard 6
 restrict-consecutive-characters english-alphabet 4 cyclic-wrap
!
```

Verification

You can use the **show aaa password-policy** command to know if the feature to restrict consecutive characters for user passwords and secrets is applied on the password policy.

```
Router#show aaa password-policy test-policy
Tue May 17 10:54:24.064 UTC
Password Policy Name : test-policy
  Number of Users : 0
  Minimum Length : 2
  Maximum Length : 253
  Special Character Len : 0
  Uppercase Character Len : 0
  Lowercase Character Len : 0
  Numeric Character Len : 0
  Policy Life Time :
    seconds : 0
    minutes : 0
    hours : 0
    days : 0
    months : 0
    years : 0
  Warning Interval :
    seconds : 0
    minutes : 0
    hours : 0
    days : 0
    months : 0
    years : 0
  Lockout Time :
    seconds : 0
    minutes : 0
    hours : 0
    days : 0
    months : 0
    years : 0
  Restrict Old Time :
    days : 0
    months : 0
    years : 0
  Character Change Len : 2
  Maximum Failure Attempts : 0
  Reference Count : 0
  Error Count : 0
  Lockout Count Attempts : 0
  Maximum char repetition : 0
```

```
  Restrict Old count : 0
  Restrict Username : 0
  Restrict Username Reverse  : 0
  Restrict Password Reverse  : 0
  Restrict Password Advanced : 0
  Restrict Consecutive Character :
    English Alphabet characters: 4
    English Alphabet Cyclic Wrap: True
    Qwerty Keyboard characters: 6
    Qwerty Keyboard Cyclic Wrap: False
Router#
```

Password or Secret Configuration Failure Scenarios:

You notice these logs or error messages on the router console when password or secret configuration fails because of the policy violation to restrict consecutive characters or numbers:

```
Router(config)#username user1
Router(config-un)#policy test-policy
Router(config-un)#password DEFg
Router(config-un)#commit
Tue Dec  7 10:17:56.843 UTC

% Failed to commit and rollback one or more configuration items. Please issue 'show
configuration failed [inheritance]' from this session to view the errors
Router(config-un)#show  configuration failed
username user1
 password 7 03205E0D01
!!% 'LOCALD' detected the 'fatal' condition 'Password contains consecutive characters from
 qwerty keyboard or English alphabet'
!
End
```

```
Router(config)#username user1
RP/0/RP0/CPU0:ios(config-un)#masked-secret
Fri Dec  3 12:33:44.354 UTC

Enter secret:
Re-enter secret:

secret is not compliant with policy to restrict consecutive letters or numbers
RP/0/RP0/CPU0:ios(config-un)#
```

```
Router(config)#username user1
Router(config-un)#policy test-policy
Router(config-un)#secret qwerty
                              ^
% Invalid input detected at '^' marker.
Router(config-un)#
```

YANG Data Model to Restrict Consecutive Characters for User Passwords and Secrets

You can use the **Cisco-IOS-XR-aaa-locald-cfg** native YANG data model to restrict consecutive characters for user passwords and secrets. **Cisco-IOS-XR-um-aaa-locald-cfg** is the corresponding unified model (UM). You can access the data models from the Github repository.

The following is a sample format to enable the feature using the native YANG data model.

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
 <edit-config>
```

```
 <target>
<candidate/>
</target>
 <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
 <aaa xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-lib-cfg">
<password-policies xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-locald-cfg">
<password-policy>
     <name>test-policy</name>
     <restrict-consecutive-characters>
      <qwerty-keyboard>
       <characters>4</characters>
 <cyclic-wrap></cyclic-wrap>
      </qwerty-keyboard>
      <english-alphabet>
       <characters>4</characters>
       <cyclic-wrap></cyclic-wrap>
      </english-alphabet>
</restrict-consecutive-characters>
</password-policy>
</password-policies>
</aaa>
</config>
</edit-config>
</rpc>
##
```

To learn more about the data models and to put them to use, see the *Programmability Configuration Guide for Cisco ASR 9000 Series Routers*.

Deprecation of Type 7 password and Type 5 secret

Password configuration options before Release 24.4.1

Until Release 24.4.1, there were two options for configuring a password:

- Password: Uses Type 7 encryption to store the password.
- Secret: Supports Type 5, 8, 9, or 10 hashing algorithms to store the password securely.

Deprecation Notice

Starting from the Release 24.4.1, the use of Type 7 password and Type 5 secret are deprecated due to security concerns. The deprecation process commences from the Release 24.4.1. We expect the full deprecation in a future release. We recommend using the default option, which is Type 10 secret.

Note With the deprecation of Type 7 password encryption in Cisco IOS XR Release 24.4.1, any configuration that used Type 7 passwords will be automatically converted and saved as Type 10 secrets during the upgrade to version 24.4.1.

If you have usernames that include both a password and a secret, then:

- For the first 3000 users, the router will retain the original secret and discard the password.
- For users beyond the first 3000, the router will convert the password as Type 10 secrets by overwriting the original secret.

password

The **password** options available in the router from the Release 24.4.1:

```
RP/0/RP0/CPU0:ios(config-un)#password ?
LINE The type 7 password followed by '7 ' OR SHA512-based password (deprecated, use 'secret')
```

Changes:

- All the options that were present until the Release 24.4.1are removed except LINE (to accept cleartext).
- **During upgrade**: Any configuration using the Type 7 password configuration is automatically converted to Type 10 secret.

  Post-upgrade: You can still use the Type 7 password configurations option after new commits, but the password will be stored as Type 10 secret.
- New **syslog** has been added to indicate the deprecation process:

  ```
  %SECURITY-PSLIB-4-DEPRECATED_PASSWORD_TYPE : The password configuration is deprecated.

   Converting it to a Type 10 secret for user <user name>.
  ```
- **show running configuration** command output before upgrade:

  ```
  username example
  password 7 106D000A0618
  !
  ```

  show running configuration command output post-upgrade:

  ```
  username example
  Cisco Confidential
  secret 10
  $6$P53pb/FFxNIT4b/.$yVakako4fp9PZiIYYh1xS0.W6b/yPrSyC8j4gLs6xli57iClOryPXyN9y8yojRD2nhAWb9pjr/WAIhbXqq8st.
  !
  ```

masked-password

The **masked-password** options available in CLI from the Release 24.4.1:

```
RP/0/RP0/CPU0:ios(config-un)#masked-password ?
0 Specifies a cleartext password will follow
clear Config deprecated. Will be removed in 7.7.1. Specify '0' instead.
<cr> The cleartext user password
```

Changes:

- The options 7 and encrypted that were present until the Release 24.4.1 are removed.
- **During upgrade**: Any configuration using the Type 7 password configuration is automatically converted to Type 10 secret.
- **Post-upgrade**: Masked-password is an alternate method of configuring the password. You can still use the masked-password keyword with a clear string after new commits, but the password will be stored as Type 10 secret.
- New **syslog** has been added to indicate the deprecation process:

  ```
  %SECURITY-PSLIB-4-DEPRECATED_PASSWORD_TYPE : The password configuration is deprecated.

   Converting it to a Type 10 secret for user <user name>.
  ```
- **show running configuration** command output before upgrade:

```
username example
password 7 106D000A0618
!
```

show running configuration command output post-upgrade:

```
username example
Cisco Confidential
secret 10
$6$P53pb/FFxNIT4b/.$yVakako4fp9PZiIYYh1xS0.W6b/yPrSyC8j4gLs6xli57iClOryPXyN9y8yojRD2nhAWb9pjr/WAIhbXqq8st.
!
```

password-policy

The **password-policy** options available in CLI from the Release 24.4.1:

```
RP/0/RP0/CPU0:ios(config-un)#password-policy ?
WORD Specify the password policy name

RP/0/RP0/CPU0:ios(config-un)#password-policy abcd password ?
0 Specifies an UNENCRYPTED password will follow
7 Specifies that an encrypted password will follow
LINE The UNENCRYPTED (cleartext) user password
clear Config deprecated. Will be removed in 7.7.1. Specify '0' instead.
encrypted Config deprecated. Will be removed in 7.7.1. Specify '7' instead.
```

Changes:

- All the options that were present until 24.4.1are removed except LINE (to accept cleartext).
- **During upgrade**: Any configuration using the Type 7 password configuration is automatically converted to Type 10 secret.

  Post-upgrade: You can still use the password-policy configurations option after new commits, but the it will be stored as Type 10 secret.
- New **syslog** has been added to indicate the deprecation process:

  ```
  %SECURITY-PSLIB-4-DEPRECATED_PASSWORD_TYPE : The password configuration is deprecated.

   Converting it to a Type 10 secret for user <username>.
  ```
- **show running configuration**f command output before upgrade:

  ```
  username example
  password-policy abcd password 7 106D000A0618
  !
  ```

  show running configuration command output post-upgrade:

  ```
  username example
  secret 10
  $6$P53pb/FFxNIT4b/.$yVakako4fp9PZiIYYh1xS0.W6b/yPrSyC8j4gLs6xli57iClOryPXyN9y8yojRD2nhAWb9pjr/WAIhbXqq8st.
  !
  !
  ```

aaa password-policy

The **aaa password-policy** options available in CLI from the Release 24.4.1:

```
RP/0/RP0/CPU0:ios(config)#aaa password-policy abcd
RP/0/RP0/CPU0:ios(config-pp)#?
min-char-change Number of characters change required between old and new passwords
(deprecated, will be removed in 25.3.1)
```

```
restrict-password-advanced Advanced restrictions on new password (deprecated, will be removed
 in 25.3.1)
restrict-password-reverse Restricts the password to be same as reversed old password
(deprecated, will be removed in 25.3.1)
```

Changes:

- The options min-char-change, restrict-password-advanced, and restrict-password-reverse that were present until the Release 24.4.1 are deprecated.
- **During upgrade**: These deprecated configurations do not go through any change during upgrade.

  Post-upgrade: These deprecated keywords do not take effect when configured post-upgrade.
- New **syslog** have been added to indicate the deprecation process:

  ```
  %SECURITY-LOCALD-4-DEPRECATED_PASSWORD_POLICY_OPTION : The password policy option
  'min-char-change' is deprecated.
  Password/Secret will not be checked against this option now.

  %SECURITY-LOCALD-4-DEPRECATED_PASSWORD_POLICY_OPTION : The password policy option
  'restrict-password-reverse' is deprecated.
   Password/Secret will not be checked against this option now.

  %SECURITY-LOCALD-4-DEPRECATED_PASSWORD_POLICY_OPTION : The password policy option
  'restrict-password-advanced' is deprecated.
  Password/Secret will not be checked against this option now.
  ```
- **show running configuration** command output before upgrade:

  ```
  aaa password-policy abcd
  lower-case 3
  min-char-change 1
  restrict-password-reverse
  restrict-password-advanced
  !
  ```

  show running configuration command output post-upgrade:

  ```
  aaa password-policy abcd
  lower-case 3
  min-char-change 1
  restrict-password-reverse
  restrict-password-advanced
  !
  ```

secret

The **secret** options available in CLI from the Release 24.4.1:

```
RP/0/RP0/CPU0:ios(config-un)#secret ?
0 Specifies a cleartext password will follow
10 Specifies that SHA512-based password will follow
8 Specifies that SHA256-based password will follow
9 Specifies that Scrypt-based password will follow
LINE The cleartext user password

RP/0/RP0/CPU0:ios(config-un)#secret 0 enc-type ?
<8-10> Specifies which algorithm to use. Only 8,9,10 supported [Note: Option ‘5’ is not
available to use from 24.4]
```

Changes:

- The options 5 and encrypted are removed.
- **During upgrade**: Configurations using Type 5 secret will remain unchanged.

Post-upgrade: Though the keyword 5 has been deprecated, you can still apply the existing configurations using Type 5 secret.

- New **syslog** has been added to indicate the deprecation process:

```
%SECURITY-LOCALD-2-DEPRECATED_SECRET_TYPE : Type 5 secret is deprecated.
Please use the 'secret' keyword with option type 10 for user.
```

- **show running configuration** command output before upgrade:

```
username example
secret 5 $1$kACo$2RtpcwyiRuRB/DhWzabfU1
!
!
```

show running configuration command output post-upgrade:

```
username example
secret 5 $1$kACo$2RtpcwyiRuRB/DhWzabfU1
!
!
```

masked-secret

The **masked-secret** options available in CLI from the Release 24.4.1:

```
RP/0/RP0/CPU0:ios(config-un)#masked-secret ?
0 Specifies a cleartext password will follow
Cisco Confidential
10 Specifies that SHA512-based password will follow
8 Specifies that SHA256-based password will follow
9 Specifies that Scrypt-based password will follow
clear Config deprecated. Will be removed in 7.7.1. Specify '0' instead.
<cr> The cleartext user password
```

Changes:

- The options 5 and encrypted are removed.
- **During upgrade**: Configurations using masked-secret with Type 5 will remain unchanged.

  Post-upgrade: Though the keyword 5 has been deprecated, you can still apply the existing configurations using Type 5 masked secret.

- New **syslog** has been added to indicate the deprecation process:

```
%SECURITY-LOCALD-2-DEPRECATED_SECRET_TYPE : Type 5 secret is deprecated.
Please use the 'secret' keyword with option type 10 for user.
```

- **show running configuration** command output before upgrade:

```
username example
secret 5 $1$kACo$2RtpcwyiRuRB/DhWzabfU1
!
!
```

show running configuration command output post-upgrade:

```
username example
secret 5 $1$kACo$2RtpcwyiRuRB/DhWzabfU1
!
!
```

Special use cases

Use case 1: Configurations using both Type 7 password and secret with 8, 9, or 10 hashing, for the same user

- **During upgrade**:
  - For the first 3000 username configurations, the password configuration will be rejected, and the secret configuration will remain unchanged.
  - For the rest of the username configurations, the original secret configuration will be rejected, and the password will be converted to Type 10 secret.
- **Post-upgrade**:
  - For a new username configured, or the username that is already present before the upgrade, the password configuration will be rejected.
  - New **syslog** has been added to indicate the deprecation process:

    ```
    %SECURITY-PSLIB-4-SECRET_CONFIG_PRESENT : The password configuration is deprecated.

    Once secret is configured, cannot use password config for user <user name> at index
     <x> now.
    ```

    where 'x' is a number representing the index.

Use case 2: Configurations using both Type 7 password and Type 5 secret, for the same user

- **During upgrade**:
  - For any username configuration, the original Type 5 secret configuration will be rejected, and the password will be converted to Type 10 secret.
- **Post-upgrade**:
  - For a new username configured, or the username that is already present before the upgrade, the password configuration will be converted to Type 10 secret.
  - New **syslog** has been added to indicate the deprecation process:

    ```
    %SECURITY-PSLIB-4-DEPRECATED_PASSWORD_TYPE : The password configuration is
    deprecated.
     Converting it to a Type 10 secret for user <username>.
    ```

Configuring Router to RADIUS Server Communication

This task configures router to RADIUS server communication.

The RADIUS host is normally a multiuser system running RADIUS server software from Cisco (CiscoSecure ACS), Livingston, Merit, Microsoft, or another software provider. Configuring router to RADIUS server communication can have several components:

- Hostname or IP address
- Authentication destination port
- Accounting destination port

- Retransmission value
- Timeout period
- Key string

RADIUS security servers are identified on the basis of their hostname or IP address, hostname and specific User Datagram Protocol (UDP) port numbers, or IP address and specific UDP port numbers. The combination of the IP address and UDP port numbers 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 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 an automatic switchover 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 they are configured.)

A RADIUS server and a Cisco router 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 router.

The timeout, retransmission, and encryption key values are configurable globally for all RADIUS servers, on a per-server basis, or in some combination of global and per-server settings. To apply these settings globally to all RADIUS servers communicating with the router, use the three unique global commands: **radius-server timeout**, **radius-server retransmit**, and **radius-server key**. To apply these values on a specific RADIUS server, use the **radius-server host** command.

You can configure a maximum of 30 global RADIUS servers.

Note You can configure both global and per-server timeout, retransmission, and key value commands simultaneously on the same Cisco network access server. If both global and per-server functions are configured on a router, the per-server timer, retransmission, and key value commands override global timer, retransmission, and key value commands.

SUMMARY STEPS

1. **configure**
2. **radius-server host** {*hostname* | *ip address*} [**auth-port** *port-number*] [**acct-port** *port-number*] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*]
3. **radius-server retransmit** *retries*
4. **radius-server timeout** *seconds*
5. **radius-server key** {**0** *clear-text-key* | **7** *encrypted-key* | *clear-text-key*}
6. **radius source-interface** *type instance* [**vrf** *vrf-id*]
7. Repeat Step 2, on page 63 through Step 6, on page 64 for each external server to be configured.
8. Use the **commit** or **end** command.
9. show radius

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**radius-server host** {*hostname* | *ip address*} [**auth-port** *port-number*] [**acct-port** *port-number*] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*] **Example:** Specifying a radius server hostname `RP/0/RSP0/CPU0:router(config)# radius-server host host1`	Specifies the hostname or IP address of the remote RADIUS server host. . • Use the **auth-port** *port-number* option to configure a specific UDP port on this RADIUS server to be used solely for authentication. • Use the **acct-port** *port-number* option to configure a specific UDP port on this RADIUS server to be used solely for accounting. • To configure the network access server to recognize more than one host entry associated with a single IP address, simply repeat this command as many times as necessary, making sure that each UDP port number is different. Set the timeout, retransmit, and encryption key values to use with the specific RADIUS host. • If no timeout is set, the global value is used; otherwise, enter a value in the range 1 to 1000. If no retransmit value is set, the global value is used; otherwise enter a value in the range 1 to 100. If no key string is specified, the global value is used. **Note** The key is a text string that must match t the key as the last item in the **radius-ser** but spaces within and at the end of the k quotation marks unless the quotation ma
Step 3	**radius-server retransmit** *retries* **Example:** `RP/0/RSP0/CPU0:router(config)# radius-server retransmit 5`	Specifies the number of times the Cisco IOS XR software searches the list of RADIUS server hosts before giving up. • In the example, the number of retransmission attempts is set to 5.
Step 4	**radius-server timeout** *seconds* **Example:**	Sets the number of seconds a router waits for a server host to reply before timing out. • In the example, the interval timer is set to 10 seconds.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config)# radius-server timeout 10`	
Step 5	**radius-server key** {**0** *clear-text-key* | **7** *encrypted-key* | *clear-text-key*} **Example:** `RP/0/RSP0/CPU0:router(config)# radius-server key 0 samplekey`	Sets the authentication and encryption key for all RADIUS communications between the router and the RADIUS daemon.
Step 6	**radius source-interface** *type instance* [**vrf** *vrf-id*] **Example:** `RP/0/RSP0/CPU0:router(config)# radius source-interface GigabitEthernet 0/3/0/1`	(Optional) Forces RADIUS to use the IP address of a specified interface or subinterface for all outgoing RADIUS packets. • The specified interface or subinterface must have an IP address associated with it. If the specified interface or subinterface does not have an IP address or is in the down state, then RADIUS reverts to the default. To avoid this, add an IP address to the interface or subinterface or bring the interface to the up state. The **vrf** keyword enables the specification on a per-VRF basis.
Step 7	Repeat Step 2, on page 63 through Step 6, on page 64 for each external server to be configured.	—
Step 8	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 9	show radius **Example:** `RP/0/RSP0/CPU0:router# show radius`	(Optional) Displays information about the RADIUS servers that are configured in the system.

What to do next

After configuring router to RADIUS server communication, configure RADIUS server groups. (See the Configuring RADIUS Server Groups, on page 71 section.)

Configuring RADIUS Dead-Server Detection

This task configures the RADIUS Dead-Server Detection feature.

The RADIUS Dead-Server Detection feature lets you configure and determine the criteria that is used to mark a RADIUS server as dead. If no criteria is explicitly configured, the criteria is computed dynamically on the basis of the number of outstanding transactions. The RADIUS dead-server detection configuration results in the prompt detection of RADIUS servers that have stopped responding. The prompt detection of nonresponding RADIUS servers and the avoidance of swamped and dead-to-live-to-dead-again servers result in less deadtime and quicker packet processing.

You can configure the minimum amount of time, in seconds, that must elapse from the time that the router 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 router booted, and there is a timeout, the time criterion is treated as though it was met.

In addition, you can configure the number of consecutive timeouts that must occur on the router 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. For example, 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-server deadtime** command specifies the time, in minutes, for which a server is marked as dead, remains dead, and, after this period, is marked alive even when no responses were received from it. When the dead criteria are configured, the servers are not monitored unless the **radius-server deadtime** command is configured

SUMMARY STEPS

1. **configure**
2. **radius-server deadtime** *minutes*
3. **radius-server dead-criteria time** *seconds*
4. **radius-server dead-criteria tries** *tries*
5. Use the **commit** or **end** command.
6. **show radius dead-criteria host** *ip address in IPv4 or IPv6 format* [**auth-port** *auth-port*] [**acct-port** *acct-port*]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	**radius-server deadtime** *minutes* **Example:** `RP/0/RSP0/CPU0:router(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 3	**radius-server dead-criteria time** *seconds* **Example:** `RP/0/RSP0/CPU0:router(config)# radius-server dead-criteria time 5`	Establishes the time for the dead-criteria conditions for a RADIUS server to be marked as dead.
Step 4	**radius-server dead-criteria tries** *tries* **Example:** `RP/0/RSP0/CPU0:router(config)# radius-server dead-criteria tries 4`	Establishes the number of tries for the dead-criteria conditions for a RADIUS server to be marked as dead.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 6	**show radius dead-criteria host** *ip address in IPv4 or IPv6 format* [**auth-port** *auth-port*] [**acct-port** *acct-port*] **Example:** `RP/0/RSP0/CPU0:router# show radius dead-criteria host 172.19.192.80`	(Optional) Displays dead-server-detection information that has been requested for a RADIUS server at the specified IP address.

Configuring Per VRF AAA

The Per VRF AAA functionality enables AAA services to be based on VPN routing and forwarding (VRF) instances. The Provider Edge (PE) or Virtual Home Gateway (VHG) communicates directly with the customer's RADIUS server, which is associated with the customer's VPN, without having to go through a RADIUS proxy. Thus, ISPs can scale their VPN offerings more efficiently, because they no longer have to use RADIUS proxies and they can provide their customers with the flexibility they demand.

New Vendor-Specific Attributes (VSAs)

The Internet Engineering Task Force (IETF) draft standard specifies a method for communicating vendor-specific information between the network access server and the RADIUS server by using the vendor-specific attribute (attribute 26). Attribute 26 encapsulates vendor-specific attributes, thereby, allowing vendors to support their own extended attributes otherwise not suitable for general use.

The Cisco IOS XR software RADIUS implementation supports one vendor-specific option 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 of the following 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 attribute-value (AV) pair defined in the Cisco RADIUS specification, and “sep” is “=” for mandatory attributes and “*” for optional attributes.

This table describes the VSAs that are now supported for Per VRF AAA.

Table 9: Supported VSAs for Per VRF AAA

VSA Name	Value Type	Description
Note The RADIUS VSAs—rad-serv, rad-serv-source-if, and rad-serv-vrf—must have the prefix “aaa:” before the VSA name.		

rad-serv	string	Indicates the IP address in IPv4 or IPv6 format, key, timeout, and retransmit number of a server and the group of the server.
		The VSA syntax follows:
		`rad-serv=a.b.c.d [key SomeKey] [auth-port X] [acct-port Y]`
		`[retransmit V] [timeout W].`
		Other than the IP address, all parameters are optional and are issued in any order. If the optional parameters are not specified, their default values are used.
		The key cannot contain any spaces; for “retransmit V,” “V” can range from 1 to 100; for “timeout W,” the “W” can range from 1 to 1000.

rad-serv-vrf	string	Specifies the name of the VRF that is used to transmit RADIUS packets. The VRF name matches the name that was specified through the **vrf** command.

This task configures RADIUS server groups per VRF. For information about configuring TACACS+ server groups per VRF, refer Configuring TACACS+ Server Groups, on page 74.

SUMMARY STEPS

1. **configure**
2. **aaa group server radius** *group-name*
3. **server-private** {*hostname* | *ip-address in IPv4 or IPv6 format*} [**auth-port** *port-number*] [**acct-port** *port-number*] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*]

4. **vrf** *vrf-name*
5. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**aaa group server radius** *group-name*
	Example:
	`RP/0/RSP0/CPU0:router(config)# aaa group server radius radgroup1`
	`RP/0/RSP0/CPU0:router(config-sg-radius)#`
		Groups different server hosts into distinct lists and enters the server group configuration mode.

Step 3	**server-private** {*hostname* | *ip-address in IPv4 or IPv6 format*} [**auth-port** *port-number*] [**acct-port** *port-number*] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*]
	Example:
	IP address in IPv4 format
	`RP/0/RSP0/CPU0:router(config-sg-radius)# server-private 10.1.1.1 timeout 5`
	`RP/0/RSP0/CPU0:router(config-sg-radius)# server-private 10.2.2.2 retransmit 3`
	Example:
	IP address in IPv6 format
	`RP/0/RSP0/CPU0:router(config-sg-radius)# server-private 2001:db8:a0b:12f0::1/64 timeout 5`
	`RP/0/RSP0/CPU0:router(config-sg-radius)# server-private 10.2.2.2 retransmit 3`
		Configures the IP address of the private RADIUS server for the group.
		If private server parameters are not specified, global configurations are used. If global configurations are not specified, default values are used.
		Both **auth-port** and **acct-port** keywords enter RADIUS server-group private configuration mode.
		You can configure a maximum of 30 private servers per RADIUS server group.

Step 4	**vrf** *vrf-name* **Example:** `RP/0/RSP0/CPU0:router(config-sg-radius)# vrf v2.44.com`	Configures the VRF reference of an AAA RADIUS server group. **Note** Private server IP addresses can overlap with distinguish them.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions:

	Command or Action	Purpose
		• **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring a TACACS+ Server

This task configures a TACACS+ server.

The port, if not specified, defaults to the standard port number, 49. The **timeout** and **key** parameters can be specified globally for all TACACS+ servers. The **timeout** parameter specifies how long the AAA server waits to receive a response from the TACACS+ server. The **key** parameter specifies an authentication and encryption key shared between the AAA server and the TACACS+ server.

The **single-connection** parameter specifies to multiplex all TACACS+ requests to the TACACS+ server over a single TCP connection. The **single-connection-idle-timeout** parameter specifies the timeout value for this single connection.

You can configure a maximum of 30 global TACACS+ servers.

SUMMARY STEPS

1. **configure**
2. **tacacs-server host** *host-name* **port** *port-number*
3. **tacacs-server host** *host-name* **timeout** *seconds*
4. **tacacs-server host** *host-name* **key** [**0** | **7**] *auth-key*
5. **tacacs-server host** *host-name* **single-connection**
6. **tacacs-server host** *host-name* **single-connection-idle-timeout** *timeout-in-seconds*
7. **tacacs source-interface** *type instance* **vrf** *vrf-name*
8. Repeat step 2 through step 6 for each external server to be configured.
9. Use the **commit** or **end** command.
10. **show tacacs**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

	Command or Action	Purpose

Step 2	**tacacs-server host** *host-name* **port** *port-number*
	Example:
	`RP/0/RSP0/CPU0:router(config)# tacacs-server host 209.165.200.226 port 51`
	`RP/0/RSP0/CPU0:router(config-tacacs-host)#`
		Specifies a TACACS+ host server and optionally specifies a server port number.
		• This option overrides the default, port 49. Valid port numbers range from 1 to 65535.

Step 3	**tacacs-server host** *host-name* **timeout** *seconds*
	Example:
	`RP/0/RSP0/CPU0:router(config-tacacs-host)# tacacs-server host 209.165.200.226 timeout 30`
	`RP/0/RSP0/CPU0:router(config)#`
		Specifies a TACACS+ host server and optionally specifies a timeout value that sets the length of time the AAA server waits to receive a response from the TACACS+ server.
		• This option overrides the global timeout value set with the **tacacs-server timeout** command for only this server. The timeout value is expressed as an integer in terms of timeout interval seconds. The range is from 1 to 1000.

Step 4	**tacacs-server host** *host-name* **key** [**0** | **7**] *auth-key* **Example:** `RP/0/RSP0/CPU0:router(config)# tacacs-server host 209.165.200.226 key 0 a_secret`	Specifies a TACACS+ host server and optionally specifies an authentication and encryption key shared between the AAA server and the TACACS+ server. • The TACACS+ packets are encrypted using this key. This key must match the key used by TACACS+ daemon. Specifying this key overrides the global key set by the **tacacs-server key** command for only this server. • (Optional) Entering **0** indicates that an unencrypted (clear-text) key follows. • (Optional) Entering **7** indicates that an encrypted key follows. • The *auth-key* argument specifies the encrypted or unencrypted key to be shared between the AAA server and the TACACS+ server.
Step 5	**tacacs-server host** *host-name* **single-connection** **Example:** `RP/0/RSP0/CPU0:router(config)# tacacs-server host 209.165.200.226 single-connection`	Prompts the router to multiplex all TACACS+ requests to this server over a single TCP connection. By default, a separate connection is used for each session.
Step 6	**tacacs-server host** *host-name* **single-connection-idle-timeout** *timeout-in-seconds* **Example:** `RP/0/0RP0RSP0/CPU0:router:hostname(config)# tacacs-server host 209.165.200.226 single-connection-idle-timeout 60`	Sets the idle timeout value, in seconds, for the single TCP connection (that is created by configuring the **single-connection** command) to the TACACS+ server. The range is: • 500 to 7200 (prior to Cisco IOS XR Software Release 6.8.1 (for 32-bit Cisco IOS XR routers) and Release 7.3.2/ Release 7.4.1 (for 64-bit Cisco IOS XR routers))

	Command or Action	Purpose
		• 5 to 7200 (starting from Cisco IOS XR Software Release 6.8.1 (for 32-bit Cisco IOS XR routers) and Release 7.3.2/ Release 7.4.1 (for 64-bit Cisco IOS XR routers))
Step 7	**tacacs source-interface** *type instance* **vrf** *vrf-name* **Example:** `RP/0/RSP0/CPU0:router(config)# tacacs source-interface GigabitEthernet 0/4/0/0 vrf abc`	(Optional) Specifies the source IP address of a selected interface for all outgoing TACACS+ packets. • The specified interface or subinterface must have an IP address associated with it. If the specified interface or subinterface does not have an IP address or is in the down state, then TACACS+ reverts to the default interface. To avoid this, add an IP address to the interface or subinterface or bring the interface to the up state. • The **vrf** option specifies the Virtual Private Network (VPN) routing and forwarding (VRF) reference of an AAA TACACS+ server group.
Step 8	Repeat step 2 through step 6 for each external server to be configured.	—
Step 9	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 10	**show tacacs** **Example:** `RP/0/RSP0/CPU0:router# show tacacs`	(Optional) Displays information about the TACACS+ servers that are configured in the system.

What to do next

After configuring TACACS+ servers, configure TACACS+ server groups. (See the Configuring TACACS+ Server Groups, on page 74 section.)

Configuring RADIUS Server Groups

This task configures RADIUS server groups.

The user can enter one or more **server** commands. The **server** command specifies the hostname or IP address of an external RADIUS server along with port numbers. When configured, this server group can be referenced from the AAA method lists (used while configuring authentication, authorization, or accounting). (See the Method Lists, on page 14 section.)

You can configure:

- Maximum 30 globally configured RADIUS server definitions.
- Maximum 30 RADIUS server groups (1global + 29 named groups) can be configured.
- Maximum 5 references to globally configured RADIUS servers in one named group.
- Maximum 30 private RADIUS servers in one named group.

Note The server and server-private limits are independent. Private servers do not consume the five-server limit for globally configured RADIUS server references.

Before you begin

For configuration to succeed, the external server should be accessible at the time of configuration.

SUMMARY STEPS

1. **configure**
2. **aaa group server radius** *group-name*
3. **server** {*hostname* | *ip address in IPv4 or IPv6 format*} [**auth-port** *port-number*] [**acct-port** *port-number*]
4. Repeat Step 4, on page 73 for every external server to be added to the server group named in Step 3, on page 73.
5. **deadtime** *minutes*
6. Use the **commit** or **end** command.
7. **show radius server-groups** [*group-name* [**detail**]]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**aaa group server radius** *group-name* **Example:** `RP/0/RSP0/CPU0:router(config)# aaa group server radius radgroup1`	Groups different server hosts into distinct lists and enters the server group configuration mode.

	Command or Action	Purpose

Step 3	**server** {*hostname* | *ip address in IPv4 or IPv6 format*} [**auth-port** *port-number*] [**acct-port** *port-number*]
	Example:
	IP address in IPv4 format
	`RP/0/RSP0/CPU0:router(config-sg-radius)# server 192.168.20.0`
	Example:
	IP address in IPv6 format
	`RP/0/RSP0/CPU0:router(config-sg-radius)# server 2001:db8:a0b:12f0::1/64`
		Specifies the hostname or IP address of an external RADIUS server.
		• After the server group is configured, it can be referenced from the AAA method lists (used while configuring authentication, authorization, or accounting).

Step 4	Repeat Step 4, on page 73 for every external server to be added to the server group named in Step 3, on page 73.	—
Step 5	**deadtime** *minutes* **Example:** `RP/0/RSP0/CPU0:router(config-sg-radius)# deadtime 1`	Configures the deadtime value at the RADIUS server group level. • The *minutes* argument specifies the length of time, in minutes, for which a RADIUS server is skipped over by transaction requests, up to a maximum of 1440 (24 hours). The range is from 1 to 1440. The example specifies a one-minute deadtime for RADIUS server group radgroup1 when it has failed to respond to authentication requests for the **deadtime** command **Note** You can configure the group-level deadt
Step 6	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 7	**show radius server-groups** [*group-name* [**detail**]] **Example:** `RP/0/RSP0/CPU0:router# show radius server-groups`	(Optional) Displays information about each RADIUS server group that is configured in the system.

What to do next

After configuring RADIUS server groups, define method lists by configuring authentication, authorization, and accounting. (See the Configuring AAA Method Lists, on page 81 section.)

Configuring TACACS+ Server Groups

This task configures TACACS+ server groups.

You can enter one or more **server** commands. The **server** command specifies the hostname or IP address of an external TACACS+ server. Once configured, this server group can be referenced from the AAA method lists (used while configuring authentication, authorization, or accounting). (See the Method Lists, on page 14 section.)

Before you begin

For successful configuration, the external server should be accessible at the time of configuration. When configuring the same IP address for global and vrf configuration, server-private parameters are required.

SUMMARY STEPS

1. **configure**
2. **aaa group server tacacs+** *group-name*
3. **server** {*hostname* | *ip address in IPv4 or IPv6 format*}
4. Repeat Step 3, on page 74 for every external server to be added to the server group named in Step 2, on page 74.
5. **server-private** {*hostname* | *ip-address in IPv4 or IPv6 format*} [**port** *port-number*] [**timeout** *seconds*] [**key** *string*]
6. **vrf** *vrf-name*
7. Use the **commit** or **end** command.
8. **show tacacs server-groups**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**aaa group server tacacs+** *group-name* **Example:** `RP/0/RSP0/CPU0:router(config)# aaa group server tacacs+ tacgroup1`	Groups different server hosts into distinct lists and enters the server group configuration mode.
Step 3	**server** {*hostname*	*ip address in IPv4 or IPv6 format*} **Example:**	Specifies the hostname or IP address of an external TACACS+ server.

	Command or Action	Purpose
	IP address in IPv4 format `RP/0/RSP0/CPU0:router(config-sg-tacacs+)# server 192.168.100.0`	• When configured, this group can be referenced from the AAA method lists (used while configuring authentication, authorization, or accounting).
Step 4	Repeat Step 3, on page 74 for every external server to be added to the server group named in Step 2, on page 74.	—
Step 5	**server-private** {*hostname* | *ip-address in IPv4 or IPv6 format*} [**port** *port-number*] [**timeout** *seconds*] [**key** *string*] **Example:** `RP/0/RSP0/CPU0:router(config-sg-tacacs+)# server-private 10.1.1.1 key a_secret`	Configures the IP address of the private TACACS+ server for the group server. **Note** • You can configure a maximum of 1 • You can configure a maximum of 1 • If private server parameters are not are not specified, default values are
Step 6	**vrf** *vrf-name* **Example:** `RP/0/RSP0/CPU0:router(config-sg-tacacs+)# vrf abc`	Specifies the Virtual Private Network (VPN) routing and forwarding (VRF) reference of an AAA TACACS+ server group.
Step 7	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 8	**show tacacs server-groups** **Example:** `RP/0/RSP0/CPU0:router# show tacacs server-groups`	(Optional) Displays information about each TACACS+ server group that is configured in the system.

What to do next

After configuring TACACS+ server groups, define method lists used by configuring authentication, authorization, and accounting. (See the Configuring AAA Method Lists, on page 81 section.)

Configure Per VRF TACACS+ Server Groups

The Cisco IOS XR software supports per VRF AAA to be configured on TACACS+ server groups. You must use the **server-private** and **vrf** commands as listed below to configure this feature.

The global server definitions can be referred from multiple server groups, but all references use the same server instance and connect to the same server. In case of VRF, you do not need the global configuration because the server status, server statistics and the key could be different for different VRFs. Therefore, you must use the server-private configuration if you want to configure per VRF TACACS+ server groups. If you have the same server used in different groups with different VRFs, ensure that it is reachable through all those VRFs.

If you are migrating the servers to a VRF, then it is safe to remove the global server configuration with respect to that server.

Prerequisites

You must ensure these before configuring per VRF on TACACS+ server groups:

- Be familiar with configuring TACACS+, AAA, per VRF AAA, and group servers.
- Ensure that you have access to the TACACS+ server.
- Configure the VRF instance before configuring the specific VRF for a TACACS+ server and ensure that the VRF is reachable.

Configuration Example

```
Router#configure

/* Groups different server hosts into distinct lists and enters the server group configuration
 mode.
You can enter one or more server commands. The server command specifies the hostname or IP
 address of an external TACACS+ server.
Once configured, this server group can be referenced from the AAA method lists (used while
 configuring authentication, authorization, or accounting). */

Router(config)# aaa group server tacacs+ tacgroup1

/* Configures the IP address and the secret key of the private TACACS+ server that is
reachable through specific VRF.
You can have multiple such server configurations which are reachable through the same VRF.*/

Router(config-sg-tacacs+)# server-private 10.1.1.1 port 49 key a_secret

/* The vrf option specifies the VRF reference of a AAA TACACS+ server group */
Router(config-sg-tacacs+)# vrf test-vrf
Router(config-sg-tacacs+)# commit
```

Running Configuration

```
aaa group server tacacs+ tacgroup1
 vrf test-vrf
 server-private 10.1.1.1 port 49
  key 7 0822455D0A16
 !
 server-private 10.1.1.2 port 49
```

```
  key 7 05080F1C2243
 !
 server-private 2001:db8:1::1 port 49
  key 7 045802150C2E
 !
 server-private 2001:db8:1::2 port 49
  key 7 13061E010803
 !
!
```

Verify Per VRF TACACS+ Server Groups

```
Router#show tacacs
Fri Sep 27 11:14:34.991 UTC

Server: 10.1.1.1/49 vrf=test-vrf [private]
        opens=0 closes=0 aborts=0 errors=0
        packets in=0 packets out=0
        status=up single-connect=false family=IPv4

Server: 10.1.1.2/49 vrf=test-vrf [private]
        opens=0 closes=0 aborts=0 errors=0
        packets in=0 packets out=0
        status=up single-connect=false family=IPv4

Server: 2001:db8:1::1/49 vrf=test-vrf [private]
        opens=0 closes=0 aborts=0 errors=0
        packets in=0 packets out=0
        status=up single-connect=false family=IPv6

Server: 2001:db8:1::2/49 vrf=test-vrf [private]
        opens=0 closes=0 aborts=0 errors=0
        packets in=0 packets out=0
        status=up single-connect=false family=IPv6
```

Associated Commands

- **server-private**
- **vrf**

View TACACS+ information in Router

Table 10: Feature History Table

Feature Name	Release Information	Feature Description
View TACACS+ information in Router	Release 7.5.4	With this feature, you can view TCP connection statistics like failures, timeout, and disconnect in connections, number of AAA packets received from an external server or sent to an external server, and so on during TACACS+ transactions. This information helps you monitor TACACS+ health in the routers. It is also helpful in identifying and debugging TACACS+ transaction failures if any. This feature introduces the following commands: • **show tacacs counters** • **show tacacs details** • **show tacacs source-interface** • **clear tacacs counters**

You can see the record of the number of requests, timeouts, failures, errors, and success for each TACACS+ server for all the AAA services using the following:

```
Router:ios# show tacacs counters

TACACS+ Server:   10.105.236.101/4010 [global]

  Authentication:
   10 requests, 4 accepts, 3 failure, 2 error, 1 timeout

  Exec Authorization:
   0 requests, 0 accepts, 0 denied, 0 error, 0 timeout

  Command Authorization:
   6 requests, 6 accepts, 0 denied, 0 error, 0 timeout

  Exec Accounting:
   0 requests, 0 accepts, 0 fail, 0 error, 0 timeout

  Command Accounting:
   6 requests, 6 accepts, 0 fail, 0 error, 0 timeout

TACACS+ Server:   10.105.236.101/2201 [private] vrf = default

  Authentication:
   0 requests, 0 accepts, 0 failure, 0 error, 0 timeout
```

```
  Exec Authorization:
   0 requests, 0 accepts, 0 denied, 0 error, 0 timeout

  Command Authorization:
   0 requests, 0 accepts, 0 denied, 0 error, 0 timeout

  Exec Accounting:
   0 requests, 0 accepts, 0 fail, 0 error, 0 timeout

  Command Accounting:
   0 requests, 0 accepts, 0 fail, 0 error, 0 timeout
```

You can view the complete TACACS+ statistics including server group, source-interface, individual server statistics, inpacket, outpacket, connection open and connection close counters, and TCP connection related counters using the following:

```
Router:ios# show tacacs details

TACACS+ Server                                                  : 10.105.236.101/4010
[Global]
    Family                                                      : IPv4
    Timeout(in secs)                                            : 3
    Connection Opens                                            : 8
    Connection Closes                                           : 8
    Requests sent                                               : 6
    Response received                                           : 6
    Packets Abort                                               : 2
    Server State                                                : Down
    Server On-Hold                                              : True
    Tacacs-Single-Connect                                       : False
    Tacacs-Single-Connect-Idle-Timeout(in secs)                 : 0
    Last Connection Attempted                                   : 08:32:43 UTC Tue Aug
02 2022


TACACS+ Server                                                  : 10.105.236.101/8010
[Private] vrf=default
    Family                                                      : IPv4
    Timeout(in secs)                                            : 3
    Connection Opens                                            : 8
    Connection Closes                                           : 7
    Requests sent                                               : 7
    Response received                                           : 7
    Packets Abort                                               : 0
    Server State                                                : Up
    Server On-Hold                                              : False
    Tacacs-Single-Connect                                       : False
    Tacacs-Single-Connect-Idle-Timeout(in secs)                 : 0
    Last Connection Attempted                                   : 08:32:52 UTC Tue Aug
02 2022


TACACS+ Server-groups:

Global list of servers
    Server 10.105.236.101/4010 family=IPv4

Server group 'tac1' has 1 servers
    Servers in this group are under 'default' vrf
    Server 10.105.236.101/8010 [private] family=IPv4

TACACS+ Source-Interface:

Interface                                           VRF Id
```

```
IPV4-Address

GigabitEthernet0/0/0/0                                  0x60000001
0.0.0.0
MgmtEth0/RP0/CPU0/0                                     0x60000000
192.168.122.222


Interface                                                VRF Id
IPV6-Address

GigabitEthernet0/0/0/0                                   0x60000001
 ::
MgmtEth0/RP0/CPU0/0                                      0x60000000
 ::
```

You can view the TACACS+ source interface details using the following:

```
Router:ios# show tacacs source-interfaces

Interface                                                VRF Id
IPV4-Address

MgmtEth0/RP0/CPU0/0                                     0x60000000
192.168.122.222

Interface                                                VRF Id
IPV6-Address

MgmtEth0/RP0/CPU0/0                                     0x60000000
::
```

You can clear all AAA services counters in **show tacacs counters** command for all TACACS+ servers using the command:

```
Router:ios# show tacacs counters

TACACS+ Server:   10.105.236.101/4010 [global]

  Authentication:
   10 requests, 4 accepts, 3 failure, 2 error, 1 timeout

  Exec Authorization:
   0 requests, 0 accepts, 0 denied, 0 error, 0 timeout

  Command Authorization:
   6 requests, 6 accepts, 0 denied, 0 error, 0 timeout

  Exec Accounting:
   0 requests, 0 accepts, 0 fail, 0 error, 0 timeout

  Command Accounting:
   6 requests, 6 accepts, 0 fail, 0 error, 0 timeout

TACACS+ Server:   10.105.236.101/2201 [private] vrf = default

  Authentication:
   0 requests, 0 accepts, 0 failure, 0 error, 0 timeout

  Exec Authorization:
   0 requests, 0 accepts, 0 denied, 0 error, 0 timeout

  Command Authorization:
   0 requests, 0 accepts, 0 denied, 0 error, 0 timeout
```

```
  Exec Accounting:
   0 requests, 0 accepts, 0 fail, 0 error, 0 timeout

  Command Accounting:
   0 requests, 0 accepts, 0 fail, 0 error, 0 timeout

Router:ios# clear tacacs counters
Router:ios# show tacacs counters

TACACS+ Server:   10.105.236.101/4010 [global]

  Authentication:
   0 requests, 0 accepts, 0 failure, 0 error, 0 timeout

  Exec Authorization:
   0 requests, 0 accepts, 0 denied, 0 error, 0 timeout

  Command Authorization:
   0 requests, 0 accepts, 0 denied, 0 error, 0 timeout

  Exec Accounting:
   0 requests, 0 accepts, 0 fail, 0 error, 0 timeout

  Command Accounting:
   0 requests, 0 accepts, 0 fail, 0 error, 0 timeout

TACACS+ Server:   10.105.236.101/2201 [private] vrf = default

  Authentication:
   0 requests, 0 accepts, 0 failure, 0 error, 0 timeout

  Exec Authorization:
   0 requests, 0 accepts, 0 denied, 0 error, 0 timeout

  Command Authorization:
   0 requests, 0 accepts, 0 denied, 0 error, 0 timeout

  Exec Accounting:
   0 requests, 0 accepts, 0 fail, 0 error, 0 timeout

  Command Accounting:
   0 requests, 0 accepts, 0 fail, 0 error, 0 timeout
```

Configuring AAA Method Lists

AAA data may be stored in a variety of data sources. AAA configuration uses *method lists* to define an order of preference for the source of AAA data. AAA may define more than one method list and applications (such as login) can choose one of them. For example, console and aux ports may use one method list and the vty ports may use another. If a method list is not specified, the application tries to use a default method list.

This section contains the following procedures:

Configuring Authentication Method Lists

This task configures method lists for authentication.

Authentication Configuration

Authentication is the process by which a user (or a principal) is verified. Authentication configuration uses *method lists* to define an order of preference for the source of AAA data, which may be stored in a variety of

data sources. You can configure authentication to define more than one method list and applications (such as login) can choose one of them. For example, console and aux ports may use one method list and the vty ports may use another. If a method list is not specified, the application tries to use a default method list.

Note Applications should explicitly refer to defined method lists for the method lists to be effective.

The authentication can be applied to tty lines through use of the **login authentication** line configuration submode command.

Creation of a Series of Authentication Methods

Use the **aaa authentication** command to create a series of authentication methods, or method list. A method list is a named list describing the authentication methods to be used (such as RADIUS or TACACS+), in sequence. The method will be one of the following:

- group radius—Use a server group or RADIUS servers for authentication
- group tacacs+—Use a server group or TACACS+ servers for authentication
- local—Use the local username or password database for authentication
- line—Use the line password or user group for authentication

If the method is RADIUS or TACACS+ servers, rather than server group, the RADIUS or TACACS+ server is chosen from the global pool of configured RADIUS and TACACS+ servers, in the order of configuration. Servers from this global pool are the servers that can be selectively added to a server group.

The subsequent methods of authentication are used only if the initial method returns an error, not if the request is rejected.

Before you begin

Note The default method list is applied for all the interfaces for authentication, except when a non-default named method list is explicitly configured, in which case the named method list is applied.

The **group radius, group tacacs+**, and **group** *group-name* forms of the **aaa authentication** command refer to a set of previously defined RADIUS or TACACS+ servers. Use the **radius server-host or tacacs-server host** command to configure the host servers. Use the **aaa group server radius or aaa group server tacacs+** command to create a named group of servers.

SUMMARY STEPS

1. **configure**
2. **aaa authentication** {**login** | **ppp**} {**default** | *list-name* | **remote**} *method-list*
3. Use the **commit** or **end** command.
4. Repeat Step 1 through Step 3 for every authentication method list to be configured.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**aaa authentication** {**login** | **ppp**} {**default** | *list-name* | **remote**} *method-list* **Example:** `RP/0/RSP0/CPU0:router(config)# aaa authentication login default group tacacs+`	Creates a series of authentication methods, or a method list. • Using the **login** keyword sets authentication for login. Using the **ppp** keyword sets authentication for Point-to-Point Protocol. • Entering the **default** keyword causes the listed authentication methods that follow this keyword to be the default list of methods for authentication. • Entering a *list-name* character string identifies the authentication method list. • Entering the **remote** keyword causes the listed authentication methods that follow this keyword to be the default list of methods for administrative authentication on a remote non-owner SDR. **Note** The remote keyword is available only c • Entering a *method-list* argument following the method list type. Method list types are entered in the preferred sequence. The listed method types are any one of the following options: • **group tacacs+**—Use a server group or TACACS+ servers for authentication • **group radius**—Use a server group or RADIUS servers for authentication • **group** *named-group*—Use a named subset of TACACS+ or RADIUS servers for authentication • **local**—Use a local username or password database for authentication • **line**—Use line password or user group for authentication

	Command or Action	Purpose
		• The example specifies the **default** method list to be used for authentication.
Step 3	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 4	Repeat Step 1 through Step 3 for every authentication method list to be configured.	—

What to do next

After configuring authentication method lists, configure authorization method lists. (See the Configuring Authorization Method Lists, on page 84 section).

Configuring Authorization Method Lists

This task configures method lists for authorization.

Note You can configure the **radius** keyword for the **aaa authorization** command.

Authorization Configuration

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

Note The software attempts authorization with the next listed method only when there is no response or an error response (not a failure) 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 type of authorization being requested. Four types of AAA authorization are supported:

- **Commands authorization—**Applies to the EXEC mode mode commands a user issues. Command authorization attempts authorization for all EXEC mode mode commands.

Note "Command" authorization is distinct from "task-based" authorization, which is based on the task profile established during authentication.

- **EXEC mode authorization—**Applies authorization for starting EXEC mode session.
- **Network authorization—**Applies authorization for network services, such as IKE.
- **Eventmanager authorization—**Applies an authorization method for authorizing an event manager (fault manager). RADIUS servers are not allowed to be configured for the event manager (fault manager) authorization. You are allowed to use TACACS+ or locald.

When you create a named method list, you are defining a particular list of authorization methods for the indicated authorization type. When defined, method lists must be applied to specific lines or interfaces before any of the defined methods are performed. Do not use the names of methods, such as TACACS+, when creating a new method list.

"Command" authorization, as a result of adding a command authorization method list to a line template, is separate from, and is in addition to, "task-based" authorization, which is performed automatically on the router. The default behavior for command authorization is none. Even if a default method list is configured, that method list has to be added to a line template for it to be used.

The **aaa authorization commands** command causes a request packet containing a series of attribute value (AV) pairs to be sent to the TACACS+ daemon as part of the authorization process. The daemon can do one of the following:

- Accept the request as is.
- Refuse authorization.

Note To avoid lockouts in user authorization, make sure to allow local fallback (by configuring the **local** option for **aaa authorization** command) when configuring AAA. For example, **aaa authorization commands default tacacs+ local**.

Creation of a Series of Authorization Methods

Use the **aaa authorization** command to set parameters for authorization and to create named method lists defining specific authorization methods that can be used for each line or interface.

The Cisco IOS XR software supports the following methods for authorization:

- **none—**The router does not request authorization information; authorization is not performed over this line or interface.
- **local—**Uses local database for authorization.

- **group tacacs+—**Uses the list of all configured TACACS+ servers for authorization.
- **group radius—**Uses the list of all configured RADIUS servers for authorization.
- **group** *group-name*—Uses a named subset of TACACS+ servers for authorization.



Note If you have configured AAA authorization to be subjected to TACACS+ authorization, then you must ensure that the server group is configured (use the **aaa group server tacacs+** command for this) for that TACACS+ server. Else, authorization fails.

For example,

```
aaa authorization exec default group test_tacacs+ local
aaa authorization commands default group test_tacacs+
aaa group server tacacs+ test_tacacs+ <===
```

SUMMARY STEPS

1. **configure**
2. **aaa authorization** {**commands** | **eventmanager** | **exec** | **network**} {**default** | *list-name*} {**none** | **local** | **group** {**tacacs+** | **radius** | *group-name*}}
3. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**aaa authorization** {**commands**	**eventmanager**	**exec**	**network**} {**default**	*list-name*} {**none**	**local**	**group** {**tacacs+**	**radius**	*group-name*}} **Example:** `RP/0/RSP0/CPU0:router(config)# aaa authorization commands listname1 group tacacs+`	Creates a series of authorization methods, or a method list. • The **commands** keyword configures authorization for all EXEC mode shell commands. Command authorization applies to the EXEC mode commands issued by a user. Command authorization attempts authorization for all EXEC modecommands. • The **eventmanager** keyword applies an authorization method for authorizing an event manager (fault manager). • The **exec** keyword configures authorization for an interactive (EXEC mode) session. • The **network** keyword configures authorization for network services like PPP or IKE.

	Command or Action	Purpose

		• The **default** keyword causes the listed authorization methods that follow this keyword to be the default list of methods for authorization.
		• A *list-name* character string identifies the authorization method list. The method list itself follows the method list name. Method list types are entered in the preferred sequence. The listed method list types can be any one of the following:
		  • **none**—The network access server (NAS) does not request authorization information. Authorization always succeeds. No subsequent authorization methods will be attempted. However, the task ID authorization is always required and cannot be disabled.
		  • **local**—Uses local database for authorization.
		• **group tacacs+**—Uses the list of all configured TACACS+ servers for authorization. The NAS exchanges authorization information with the TACACS+ security daemon. TACACS+ authorization defines specific rights for users by associating AV pairs, which are stored in a database on the TACACS+ security server, with the appropriate user.
		• **group radius**—Uses the list of all configured RADIUS servers for authorization.
		• **group** *group-name*—Uses a named server group, a subset of TACACS+ or RADIUS servers for authorization as defined by the **aaa group server tacacs+** or **aaa group server radius** command.

Step 3	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

After configuring authorization method lists, configure accounting method lists. (See the Configuring Accounting Method Lists, on page 88 section.)

Configuring Accounting Method Lists

This task configures method lists for accounting.

Note You can configure the **radius** keyword for the **aaa accounting** command.

Accounting Configuration

Currently, Cisco IOS XR software supports both the TACACS+ and RADIUS methods for accounting. The router reports user activity to the TACACS+ or RADIUS security server in the form of accounting records. Each accounting record contains accounting AV pairs and is stored on the security server.

Method lists for accounting define the way accounting is performed, enabling you to designate a particular security protocol to be used on specific lines or interfaces for particular types of accounting services. When naming a method list, do not use the names of methods, such as TACACS+.

For minimal accounting, include the **stop-only** keyword to send a "stop accounting" notice at the end of the requested user process. For more accounting, you can include the **start-stop** keyword, so that the external AAA server sends a "start accounting" notice at the beginning of the requested process and a "stop accounting" notice at the end of the process. In addition, you can use the **aaa accounting update** command to periodically send update records with accumulated information. Accounting records are stored only on the TACACS+ or RADIUS server.

When AAA accounting is activated, the router reports these attributes as accounting records, which are then stored in an accounting log on the security server.

Creation of a Series of Accounting Methods

Use the **aaa accounting** command to create default or named method lists defining specific accounting methods that can be used for each line or interface.

The Cisco IOS XR software supports the following methods for accounting:

- none—Accounting is not performed over this line or interface.
- group tacacs+—Use the list of all configured TACACS+ servers for accounting.
- group radius—Use the list of all configured RADIUS servers for accounting.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
   - **aaa accounting** {**commands** | **exec** | **network**} {**default** | *list-name*} {**start-stop** | **stop-only**}
   - {none | method}
3. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	Do one of the following: • **aaa accounting** {**commands** | **exec** | **network**} {**default** | *list-name*} {**start-stop** | **stop-only**} • {none | method} **Example:** `RP/0/RSP0/CPU0:router(config)# aaa accounting commands default stop-only group tacacs+`	**Note** Command accounting is not supported o Creates a series of accounting methods, or a method list. • The **commands** keyword enables accounting for EXEC mode shell commands. • The **exec** keyword enables accounting for an interactive (EXEC mode) session. • The **network** keyword enables accounting for all network-related service requests, such as Point-to-Point Protocol (PPP). • The **default** keyword causes the listed accounting methods that follow this keyword to be the default list of methods for accounting. • A *list-name* character string identifies the accounting method list. • The **start-stop** keyword sends a "start accounting" notice at the beginning of a process and a "stop accounting" notice at the end of a process. The requested user process begins regardless of whether the "start accounting" notice was received by the accounting server. • The **stop-only** keyword sends a "stop accounting" notice at the end of the requested user process. • The **none** keyword states that no accounting is performed. • The method list itself follows the **start-stop** keyword. Method list types are entered in the preferred sequence. The method argument lists the following types: • **group tacacs+**—Use the list of all configured TACACS+ servers for accounting.

	Command or Action	Purpose
		• **group radius—**Use the list of all configured RADIUS servers for accounting. • **group** *group-name*—Use a named server group, a subset of TACACS+ or RADIUS servers for accounting as defined by the **aaa group server tacacs+** or **aaa group server radius** command. • The example defines a **default** command accounting method list, in which accounting services are provided by a TACACS+ security server, with a stop-only restriction.
Step 3	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

After configuring method lists, apply those method lists. (See the Applying Method Lists for Applications, on page 92 section.)

Generating Interim Accounting Records

This task enables periodic interim accounting records to be sent to the accounting server. When the **aaa accounting update** command is activated, Cisco IOS XR software issues interim accounting records for all users on the system.

Note Interim accounting records are generated only for network sessions, such as Internet Key Exchange (IKE) accounting, which is controlled by the **aaa accounting** command with the **network** keyword. System, command, or EXEC accounting sessions cannot have interim records generated.

SUMMARY STEPS

1. **configure**
2. **aaa accounting update** {**newinfo** | **periodic** *minutes*}
3. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**aaa accounting update** {**newinfo** | **periodic** *minutes*} **Example:** `RP/0/RSP0/CPU0:router(config)# aaa accounting update periodic 30`	Enables periodic interim accounting records to be sent to the accounting server. • If the **newinfo** keyword is used, interim accounting records are sent to the accounting server every time there is new accounting information to report. An example of this report 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 **periodic** keyword, interim accounting records are sent periodically as defined by the argument number. The interim accounting record contains all the accounting information recorded for that user up to the time the interim accounting record is sent. **Caution** The **periodic** keyword causes heavy con
Step 3	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Applying Method Lists for Applications

After you configure method lists for authorization and accounting services, you can apply those method lists for applications that use those services (console, vty, auxiliary, and so on). Applying method lists is accomplished by enabling AAA authorization and accounting.

This section contains the following procedures:

Enabling AAA Authorization

This task enables AAA authorization for a specific line or group of lines.

Method List Application

After you use the **aaa authorization** command to define a named authorization method list (or use the default method list) for a particular type of authorization, you must apply the defined lists to the appropriate lines in order for authorization to take place. Use the **authorization** command to apply the specified method lists (or, if none is specified, the default method list) to the selected line or group of lines.

SUMMARY STEPS

1. **configure**
2. **line** {**aux** | **console** | **default** | **template** *template-name*}
3. **authorization** {**commands** | **exec**} {**default** | *list-name*}
4. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**line** {**aux**	**console**	**default**	**template** *template-name*} **Example:** `RP/0/RSP0/CPU0:router(config)# line console`	Enters line template configuration mode.
Step 3	**authorization** {**commands**	**exec**} {**default**	*list-name*} **Example:** `RP/0/RSP0/CPU0:router(config-line)# authorization commands listname5`	Enables AAA authorization for a specific line or group of lines. • The **commands** keyword enables authorization on the selected lines for all commands. • The **exec** keyword enables authorization for an interactive (EXEC mode) session.

	Command or Action	Purpose
		• Enter the **default** keyword to apply the name of the default method list, as defined with the **aaa authorization** command. • Enter the name of a list of authorization methods to use. If no list name is specified, the system uses the default. The list is created with the **aaa authorization** command. • The example enables command authorization using the method list named listname5.
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

After applying authorization method lists by enabling AAA authorization, apply accounting method lists by enabling AAA accounting. (See the Enabling Accounting Services, on page 93 section.)

Enabling Accounting Services

This task enables accounting services for a specific line of group of lines.

SUMMARY STEPS

1. **configure**
2. **line { aux | console | default | template template-name**}
3. **accounting** {**commands** | **exec**} {**default** | *list-name*}
4. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router# configure`	
Step 2	**line { aux | console | default | template template-name}** **Example:** `RP/0/RSP0/CPU0:router(config)# line console`	Enters line template configuration mode.
Step 3	**accounting** {**commands** | **exec**} {**default** | *list-name*} **Example:** `RP/0/RSP0/CPU0:router(config-line)# accounting commands listname7`	Enables AAA accounting for a specific line or group of lines. • The **commands** keyword enables accounting on the selected lines for all EXEC mode shell commands. • The **exec** keyword enables accounting for an interactive (EXEC mode) session. • Enter the **default** keyword to apply the name of the default method list, as defined with the **aaa accounting** command. • Enter the name of a list of accounting methods to use. If no list name is specified, the system uses the default. The list is created with the **aaa accounting** command. • The example enables command accounting using the method list named listname7.
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

After applying accounting method lists by enabling AAA accounting services, configure login parameters. (See the Configuring Login Parameters, on page 94 section.)

Configuring Login Parameters

This task sets the interval that the server waits for reply to a login.

SUMMARY STEPS

1. **configure**
2. **line template** *template-name*
3. **timeout login response** *seconds*
4. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**line template** *template-name* **Example:** `RP/0/RSP0/CPU0:router(config)# line template alpha`	Specifies a line to configure and enters line template configuration mode.
Step 3	**timeout login response** *seconds* **Example:** `RP/0/RSP0/CPU0:router(config-line)# timeout login response 20`	Sets the interval that the server waits for reply to a login. • The *seconds* argument specifies the timeout interval (in seconds) from 0 to 300. The default is 30 seconds. • The example shows how to change the interval timer to 20 seconds.
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

How to Configure Hold-Down Timer for TACACS+

By default, the hold-down timer for TACACS+ is disabled. To enable the hold-down timer, use the **holddown-time** command under respective configuration modes as per the following hierarchy levels:

- **Global Level**: Applicable to all TACACS+ servers that are configured on the router.

- **Server Group Level**: Applicable only to TACACS+ servers that are configured in a particular server group. This configuration overrides the global hold-down timer configuration.
- **Server Level**: Applicable only to a particular TACACS+ server (that also includes the private server). This configuration overrides the timer value at all other levels.
- **Private Server Level**: Applicable only to a particular private TACACS+ server.

While selecting the timer at various configuration levels, the router gives preference to the one which is more specific to the server. That is, the server-level timer has the highest precedence, followed by server group-level and finally, the global-level timer.

Guidelines for Configuring Hold-Down Timer for TACACS+

- You must configure the TACACS+ servers for this feature to take effect.
- A timer value of zero indicates that the feature is disabled.
- The timer value is decided by the configuration that is closest to the server regardless of its value. That is, if the server-level timer is configured as 0, the system disables the feature for that particular server, even if a positive value exists at other levels. So, if you need to disable the feature for some servers or server-groups, and not for others, you can configure a zero value for those specific servers or server-groups, and configure a positive value at the global level.
- The system assigns priority to the servers based on the order in which they are configured in the router. The server that is configured first is used first. If the first server becomes unavailable or unreachable, the second server is used, and so on.
- Avoid configuring a large timer value, as it marks the server as being down for a longer period. Also, the router does not use that server for further client requests during the hold-down time, even if the server becomes available in between. As a result, we recommend that you configure an optimal timer value of say, one or two minutes.
- If there is a process restart or router reload while the timer is running, the timer immediately expires, and the router considers the unresponsive server as being up.

Syslog for Hold-Down Timer

The TACACS+ hold-down timer feature introduces a new syslog to notify that the server is marked as being down, and that the hold-down timer has started. This syslog replaces the old syslog which was invoked during earlier scenarios when server was down. If the feature is not enabled, the router continues to display the old syslog.

The syslog without enabling hold-down timer:

```
RP/0/RP0/CPU0:Aug 21 17:42:49.664 UTC: tacacsd[1226]: %SECURITY-TACACSD-6-SERVER_DOWN :
TACACS+ server 10.10.10.2/2020 is DOWN [vrf: 0x60000000, server-private: No]- Socket 116:
No route to host
```

The syslog with hold-down timer enabled:

```
RP/0/RP0/CPU0:ios#RP/0/RP0/CPU0:Aug 21 16:00:25.200 UTC: tacacsd[1227]:
%SECURITY-TACACSD-6-HOLDDOWN_TIME_START :
TACACS+ server 10.105.236.103/2020 is DOWN [vrf: 0x60000000, server-private: Yes]. Server
will be marked as DOWN for 20 seconds: Success
```

Configuration Example

- **Global Level**:

```
Router#configure
Router(config)#tacacs-server holddown-time 30
```

- **Server Level**:

```
Router(config)#tacacs-server host 10.105.236.102 port 2020
Router(config-tacacs-host)#holddown-time 35
```

- **Server-Group Level**:

```
Router#configure
Router(config)#aaa group server tacacs+ test-group
Router(config-sg-tacacs)#holddown-time 40
```

- **Private Server Level**:

```
Router(config)#aaa group server tacacs+ test-group
Router(config-sg-tacacs)#server-private 10.105.236.109 port 2020
Router(config-sg-tacacs-private)#holddown-time 55
```

Running Configuration

```
Router#show running-config
!
tacacs-server holddown-time 30
!
tacacs-server host 10.105.236.102 port 2020
 holddown-time 35
!
aaa group server tacacs+ test-group
 holddown-time 40
 server-private 10.105.236.109 port 2020
  holddown-time 55
 !
!
```

How to Disable Hold-Down Timer for TACACS+

You can disable the hold-down timer for TACACS+ at respective levels either by using the **no** form of the **holddown-time** command, or by configuring a timer value of zero.

For example,

```
Router(config)#no tacacs-server holddown-time 30
OR
Router(config)#tacacs-server holddown-time 0
```

Verification

A new field, **on-hold**, is introduced in the output field of the **show tacacs** command to indicate whether a server is on hold due to the hold-down timer or the server probe is in progress. A value of *true* indicates that the server is marked as being down. The router does not use that server for addressing any client request.

```
Router#show tacacs
Wed Oct 21 06:45:38.341 UTC
Server: 10.105.236.102/2020 opens=1 closes=1 aborts=1 errors=0
       packets in=0 packets out=0
       status=down single-connect=false family=IPv4
       idle-timeout=0 on-hold=true

Server: 10.105.236.103/2020 vrf=default [private]
        opens=0 closes=0 aborts=0 errors=0
        packets in=0 packets out=0
        status=up single-connect=false family=IPv4
        on-hold=true
```

The following is a sample output with **on-hold** value as *false*, which indicates that the server is not marked as being down. The router considers that server as being available for addressing client requests.

```
Router#show tacacs
Fri Aug 21 15:57:02.139 UTC

Server: 10.105.236.102/2020 opens=0 closes=0 aborts=0 errors=0
        packets in=0 packets out=0
        status=up single-connect=false family=IPv4
        idle-timeout=0 on-hold=false

Server: 10.105.236.103/2020 vrf=default [private]
        opens=0 closes=0 aborts=0 errors=0
        packets in=0 packets out=0
        status=up single-connect=false family=IPv4
        on-hold=false
```

Related Topics

- Hold-Down Timer for TACACS+, on page 34

Associated Commands

- **holddown-time**

Configuration Examples for Configuring AAA Services

This section provides the following configuration example:

Configuring AAA Services: Example

The following examples show how to configure AAA services.

An authentication method list vty-authen is configured. This example specifies a method list that uses the list of all configured TACACS+ servers for authentication. If that method fails, the local username database method is used for authentication.

```
configure
aaa authentication login vty-authen group tacacs+ local
```

The default method list for PPP is configured to use local method.

```
aaa authentication ppp default local
```

A username user1 is created for login purposes, a secure login password is assigned, and user1 is made a root-lr user. Configure similar settings for username user2.

```
username user1
secret lab
group root-lr
exit

username user2
secret lab
exit
```

A task group named tga is created, tasks are added to tga, a user group named uga is created, and uga is configured to inherit permissions from task group tga. A description is added to task group uga.

```
taskgroup tga
task read bgp
task write ospf
exit

usergroup uga
taskgroup tga
description usergroup uga
exit
```

Username user2 is configured to inherit from user group uga.

```
username user2
group uga
exit
```

Three TACACS servers are configured.

```
tacacs-server host 10.1.1.1 port 1 key abc
tacacs-server host 10.2.2.2 port 2 key def
tacacs-server host 10.3.3.3 port 3 key ghi
```

A user group named priv5 is created, which will be used for users authenticated using the TACACS+ method and whose entry in the external TACACS+ daemon configuration file has a privilege level of 5.

```
usergroup priv5
taskgroup operator
exit
```

An authorization method list, vty-author, is configured. This example specifies that command authorization be done using the list of all configured TACACS+ servers.

```
aaa authorization commands vty-author group tacacs+
```

An accounting method list, vty-acct, is configured. This example specifies that start-stop command accounting be done using the list of all configured TACACS+ servers.

```
aaa accounting commands vty-acct start-stop group tacacs+
```

For TACACS+ authentication, if, for example, a privilege level 8 is returned, and no local usergroup priv8 exists and no local user with the same name exists, the **aaa default-taskgroup** command with tga specified as the *taskgroup-name* argument ensures that such users are given the taskmap of the task group tga.

```
aaa default-taskgroup tga
```

For line template vty, a line password is assigned that is used with line authentication and makes usergroup uga the group that is assigned for line authentication (if used), and makes vty-authen, vty-author, and vty-acct, respectively, the method lists that are used for authentication, authorization, and accounting.

```
line template vty
password lab
users group uga
login authentication vty-authen
authorization commands vty-author
accounting commands vty-acct
exit
```

A TACACS+ server group named abc is created and an already configured TACACS+ server is added to it.

```
aaa group server tacacs+ abc
server 10.3.3.3
exit
```

TACACS+ with TLS protection

The TACACS+ with TLS protection is a security enhancement to the TACACS+ protocol that

- uses Transport Layer Security (TLS) to encrypt authentication, authorization, and accounting (AAA) communication between network devices and TACACS+ servers,
- provides confidentiality and integrity for sensitive data transmitted over potentially insecure networks, and
- supports mutual authentication to safeguard against unauthorized access.

Table 11: Feature History Table

Feature Name	Release Information	Feature Description
TACACS+ with TLS protection	Release 25.3.1	You can significantly enhance security and reduce the risk of attacks on weak encryption by using TACACS+ over TLS. This method ensures the secure transmission of all Authentication, Authorization, and Accounting (AAA) data between the client and server. It provides robust protection for sensitive environments by supporting mutual authentication through a TLS X.509 certificate-based infrastructure. This feature is compatible with both TLS versions 1.3 and 1.2.

Benefits of TACACS+ with TLS protection

- Enhances security by encrypting TACACS+ communications.

- Protects AAA data with robust encryption.
- Supports mutual authentication between client and server.
- Complies with TLS 1.3 and 1.2 standards.

TACACS+ with TLS protection improves the security of AAA services by using TLS for encrypted communication. It addresses vulnerabilities associated with weak encryption and is designed for sensitive environments that require strong protection for AAA data.

How TACACS+ with TLS protection work

Summary

TACACS+ with TLS protection secures AAA communications between a network device and a TACACS+ server by carrying TACACS+ packets within an encrypted TLS session that is established when a user initiates SSH access.

The key components involved in the process are:

- End user: Initiates an SSH session to access the network device.
- Router as TACACS+ TLS client: The router receives the AAA service request and initiates a TLS session to the TACACS+ server.
- TACACS+ server: The server uses valid TLS configuration, terminates the TLS session, and processes AAA requests and responses.
- TLS session: Encrypts the TACACS+ traffic between the client and server.
- TACACS+ packets: Carry authentication, authorization, and accounting information over the TLS tunnel.

Workflow

These are the stages of how TACACS+ with TLS protection works:

1. Session initiation: The end user initiates an SSH session to the network device.
2. AAA request: The router receives the user's AAA service request for TACACS+ with TLS protection.
3. TLS establishment: If the TACACS+ server is configured with valid TLS settings, the TACACS+ TLS client and server establish a TLS session.
4. Secure exchange: The client and server exchange TACACS+ packets over the TLS-encrypted session.

Result

The process establishes an encrypted channel for TACACS+ traffic, securing AAA communications during SSH access.

Guidelines for TACACS+ with TLS protection

Follow these guidelines when configuring TACACS+ with TLS protection:

- Configure the destination port for TACACS+ with TLS protection. There are no dedicated ports for authentication, accounting, or authorization changes.
- Use TLS X.509 certificate-based mutual authentication between client and server.
- Ensure you use either TLS 1.3 (as mandated by RFC 8446) or TLS 1.2, which is also supported.
- Use either multi-connect or single connect without TLS resumption as needed.
- Ensure your implementation supports the cipher suites mandated by TLS 1.3 and TLS 1.2.
- Use IPv4 or IPv6 for TACACS+ transactions as appropriate.
- Use the source interface and non-default Virtual Routing and Forwarding (VRF) as required.
- Configure the connection timer and single-connect idle timeout as appropriate—these settings work the same as for TACACS+ over TCP.

Restrictions for TACACS+ with TLS protection

These restrictions apply when configuring TACACS+ with TLS protection:

- The TACACS+ encryption method supported in Cisco IOS-XR software releases before 25.3.1 is no longer supported.
- The router does not support using both non-TLS and TLS servers in the same server group.
- The router does not support TLS session resumption or TLS sessions with PSK cipher suites.

Configure TACACS+ with TLS protection

Configure TACACS+ to use Transport Layer Security (TLS) for secure communication, enhancing the confidentiality and integrity of authentication, authorization, and accounting traffic.

This task describes how to enable TLS for TACACS+ server communication. You can enable TLS directly for a TACACS+ host or within an AAA server group. With TLS enabled, data exchanged between the network device and the TACACS+ server is encrypted.

Procedure

Step 1 Use the **tls** to enable TACACS+ with TLS protection.

a) Tacacs-server host configuration:

Example:

```
Router(config)# tacacs-server host 10.105.236.101 port 4950
Router(config-tacacs-host)# tls
Router(config-tacacs-host-tls)# server-name-indicator aaa.cisco.com
Router(config-tacacs-host-tls)# trustpoint test
Router(config-tacacs-host-tls)# commit
```

b) Server-group configuration:

Example:

```
Router(config)# configure
Router(config)# aaa group server tacacs+ tac1
Router(config-sg-tacacs)# server-private 10.105.236.101 port 2345
Router(config-sg-tacacs-private)# tls
Router(config-sg-tacacs-private-tls)# server-name-indicator aaa.cisco.com
Router(config-sg-tacacs-private-tls)# trustpoint abc
Router(config-sg-tacacs-private-tls)# commit
```

Step 2 Run the **show tacacs** command to display the TACACS information.

Example:

```
Router# show tacacs
Info: Verify that TACACS+ with TLS protection is configured.

Server: 10.105.236.101/2084
.....
FIPS mode : TRUE/FALSE.
TLS:
   Version: TLS 1.3/1.2      // only for active connection
   Cipher: TLS_AES_128_GCM_SHA256  // only for active connection
Statistics:
   Successfull connections: 0
   Failed connections         : 0
   SSL  errors :
    Connect error:
    Read error:
    Write error:
    Handshake Failure:
    Protocol Mismatch :
  Certificate Validation Error:
  Cipher Suite Mismatch:
  Session Timeout:
  Revoked Certificate:
  Unsupported TLS Version:
  Untrusted CA:
```

TACACS+ communication with the specified server is now secured using TLS, encrypting authentication, authorization, and accounting traffic.

Command Accounting

Command accounting with a method as local, enables the logging of commands executed by all users as syslog messages. This feature can be enabled or disabled only by users who have AAA write permissions. Once enabled, all the commands that are executed by all users can be viewed from the output of the **show logging** command.

Command accounting is not supported for commands that are executed using Netconf, XML or GRPC. Command accounting is not used as a failover accounting method but as an additional method of accounting. So this feature will be active even when other accounting methods are configured and functional.

Configuring Command Accounting

Command Accounting can either be configured alone or along with other accounting methods as shown below:

1. Configuring command accounting alone

```
RP/0/RSP0/CPU0:router(config)# aaa accounting commands default start-stop local none
RP/0/RSP0/CPU0:router(config)# commit
```

2. Configuring command accounting along with other accounting methods

```
RP/0/RSP0/CPU0:router(config)# aaa accounting commands default start-stop group tacacs+
 local none
RP/0/RSP0/CPU0:router(config)# commit
```

Model-based AAA

Table 12: Feature History Table

Feature Name	Release Information	Description
NETCONF Access Control Model (NACM) for Protocol Operations and Authorization	Release 7.4.1	NACM is defined in AAA subsystem to manage access control for NETCONF Remote Procedure Calls (RPCs). NACM addresses the need to authenticate the user or user groups, authorize whether the user has the required permission to perform the operation. With this feature, you can configure the authorization rules, groups and rule lists containing multiple groups and rules using CLI commands in addition to existing support for YANG data models. This feature also introduces `Cisco-IOS-XR-um-aaa-nacm-cfg.yang` unified data model to configure user access and privileges. You can access this data model from the Github repository.

The Network Configuration Protocol (NETCONF) protocol does not provide any standard mechanisms to restrict the protocol operations and content that each user is authorized to access. The NETCONF Access Control Model (NACM) is defined in AAA subsystem to manage access-control for NETCONF/YANG RPC requests.

The NACM module provides the ability to control the manageability activities of NETCONF users on the router. You can manage access privileges, the kind of operations that users can perform, and a history of the operations that were performed on the router. The NACM functionality accounts for all the operations that are performed on the box over the NETCONF interface. This functionality authenticates the user or user groups and authorizes permissions for users to perform the operation.

Prerequisites for Model Based AAA

Working with the model based AAA feature requires prior understanding of the following :

- NETCONF-YANG
- RFC 6536: Network Configuration Protocol (NETCONF) Access Control Model

Initial Operation

These are the NACM default values. By default a user is denied write permission, hence you'll not be able to edit the NACM configurations after enabling NACM authorization using AAA command.

```
<enable-nacm>false</enable-nacm>
<read-default>permit</read-default>
<write-default>deny</write-default>
<exec-default>permit</exec-default>
<enable-external-groups>true</enable-external-groups>
```

Therefore we recommend to enable NACM after configuring the required NACM configurations, or after changing the default NACM configurations. Here are few sample configurations:



Note If `access-denied` message is returned while writing NACM configurations, then NACM authorization can be disabled to edit the NACM configurations.

```
<aaa xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-lib-cfg">
<usernames xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-locald-cfg">
<username>
<ordering-index>3</ordering-index>
<name>username</name>
<password>password</password>
     <usergroup-under-usernames>
      <usergroup-under-username>
       <name>root-lr</name>
      </usergroup-under-username>
      <usergroup-under-username>
       <name>cisco-support</name>
      </usergroup-under-username>
     </usergroup-under-usernames>
</username>
</usernames>
</aaa>

<nacm xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-nacm-cfg">
<read-default>permit</read-default>
<write-default>permit</write-default>
<exec-default>permit</exec-default>
<enable-external-groups>true</enable-external-groups>
<groups>
    <group>
    <name>nacm_group</name>
    <user-name>lab</user-name>
    </group>
</groups>
<rule-list>
<name>Rule-list-1</name>
<group>Group_nacm_0_test</group>
```

```
<rule>
    <name>Rule-1</name>
    <access-operations>read</access-operations>
    <action>permit</action>
    <module-name>ietf-netconf-acm</module-name>
    <rpc-name>edit-config</rpc-name>
                <access-operations>*</access-operations>
                <path>/</path>
                <action>permit</action>
            </rule>
       </rule-list>
</nacm>
```

The NACM configuration allows to choose the precedence of external groups over the local groups.

NACM Configuration Management and Persistence

The NACM configuration can be modified using NETCONF or RESTCONF. In order for a user to be able to acccess the NACM configuration, they must have explicit permission to do so, that is, through a NACM rule. Configuration under the /nacm subtree persists when the **copy running-config startup-config** EXEC command is issued, or the **cisco-ia:save-config** RPC is issued.

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<save-config xmlns="http://cisco.com/yang/cisco-ia"/>
</rpc>
```

Overview of Configuring NACM

Here are the steps involved in configuring NACM:

1. Configure all NACM rules
2. Enable NACM
3. Disconnect all active NETCONF sessions
4. Launch new NETCONF session



Note Enabling or disabling NACM does not affect any existing NETCONF sessions.

NACM Rules

As per the RFC 6536, NACM defines two categories of rules:

- Global Rules—It includes the following:
  - Enable/Disable NACM
  - Read-Default
  - Write-Default
  - Exec-Default

- Enable External Groups

- Access Control Rules—It includes the following:
  - Module (used along with protocol rule / data node rule)
  - Protocol
  - Data Node

The following table lists the rules and access operations:

Operation	Description
all	Rule is applied to all types of protocol operations
create	Rule is applied to all protocol operations, which create a new data node such as edit-config operation
read	Rule is applied to all protocol operations, which reads the data node such as get, get-config or notification
update	Rule is applied to all protocol operations, which alters a data node such as edit-config operation
exec	Rule is applied to all exec protocol access operations such as action RPC
delete	Rule is applied to all protocol operations that removes a data node



Note Before enabling NACM using NETCONF RPC, any user with access to the system can create NACM groups and rules. However, after NACM is enabled, only authorised users can change the NACM configurations.



Note Only users who belong to `root-lr` group or with write access in `aaa task` group can enable or disable NACM using CLI commands.

Example: Configure Global Rules

YANG Data Model: You must configure NACM groups and NACM rulelist before configuring NACM rules. The following sample configuration shows a NACM group configuration:

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" >
<edit-config>
  <target><candidate/></target>
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
    <groups>
      <group>
        <name>group1</name>
        <user-name>user1</user-name>
```

```
        <user-name>user2</user-name>
        <user-name>user3</user-name>
      </group>
    </groups>
  </nacm>
</config>
</edit-config>
</rpc>
```

The following sample configuration shows a NACM rule list configuration:

```
<rpc
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"message-id="101">
<edit-config>
  <target>
    <candidate/>
  </target>
 <config>
  <nacm xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-nacm-cfg">
  <rulelist-classes>
  <rulelist-class>
  <ordering-index>1</ordering-index>
 <rulelist-name>GlobalRule</rulelist-name>
 <group-names>
  <group-name>root-system</group-name>
   <group-name>AdminUser</group-name>
 </group-names>
 </rulelist-class>
 </rulelist-classes>
</nacm>
</config>
</edit-config>
</rpc>
```

You can configure the NACM rule list using CLI commands in addition to configuring using YANG data models. The following commands are supported:

```
Router(config)#nacm rule-list 1 GlobalRule
Router(config-rlst)#groupnames root-system AdminUser
```

Example: Configure NACM Global Rules

YANG Data Model:

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" >
<edit-config>
  <target><candidate/></target>
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
    <read-default>permit</read-default>
    <write-default>permit</write-default>
    <exec-default>permit</exec-default>
    <enable-external-groups>false</enable-external-groups>
  </nacm>
</config>
</edit-config>
</rpc>
```

CLI Command: You can configure the NACM global rules using CLI commands in addition to configuring using YANG data models. The following commands are supported:

```
Router(config)#nacm read-default [ permit | deny ]
Router(config)#nacm write-default [ permit | deny ]
```

```
Router(config)#nacm exec-default [ permit | deny ]
Router(config)#nacm enable-external-groups [ true | false ]
```

Note You must have NACM task permissions to make changes.

Example: Configure Access Control Rules

YANG Data Model:

```
<rpc message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" >
<edit-config>
<target><candidate/></target>
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
    <rule-list>
     <name>GlobalRule</name>
     <rule>
       <name>rule1</name>
       <module-name>ietf-netconf-acm</module-name>
       <rpc-name>edit-config</rpc-name>
       <access-operations>*</access-operations>
       <action>permit</action>
     </rule>
    <rule>
      <name>rule2</name>
      <module-name>ietf-netconf-acm</module-name>
      <rpc-name>get-config</rpc-name>
      <access-operations>create read update exec</accessoperations>
      <action>permit</action>
    </rule>
   </rule-list>
  </nacm>
 </config>
 </edit-config>
</rpc>
```

Note '*' refers to all operations.

CLI Command: You can onfigure the NACM protocol rules using CLI commands in addition to configuring using YANG data models:

```
Router(config)#nacm rule-list 1 GlobalRule
Router(nacm-rlst)#groupnames AdminUser
Router(nacm-rlst)#rule 1 rule1
Router(nacm-rule)#action permit
Router(nacm-rule)#module-name ietf-netconf-acm
Router(nacm-rule)#rule-type rpc edit-config
Router(nacm-rule)#access-operations create read update exec
Router(nacm-rlst)#rule 2 rule2
Router(nacm-rule)#action deny
Router(nacm-rule)#module-name ietf-netconf-acm
Router(nacm-rule)#rule-type rpc get-config
Router(nacm-rule)#access-operations create read update exec
```

Example: Configure NACM Data Node Rules

```
<rpc message-id="101"xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" >
<edit-config>
<target><candidate/></target>
  <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
    <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
     <rule-list>
        <name>GlobalRule</name>
      <rule>
       <name>rule4</name>
       <module-name>*</module-name>
       <path>/nacm/groups/group</path>
       <access-operations>*</access-operations>
       <action>permit</action>
      </rule>
      <rule>
       <name>rule5</name>
       <module-name>ietf-netconf-acm</module-name>
       <path>/nacm/rule-list</path>
       <access-operations>read</access-operations>
       <action>deny</action>
      </rule>
     </rule-list>
   </nacm>
  </config>
 </edit-config>
</rpc>
```

Note '*' refers to all modules, and all operations.

CLI Command: You can configure the NACM data rules using CLI commands in addition to configuring using YANG data models. The following commands are supported:

```
nacm rule-list 1 GlobalRule
 groupnames AdminUser
  rule 4 rule4
   action permit
   module-name *
   rule-type data-node /nacm/groups/group
   access-operations all
  rule 5 rule5
   action deny
   module-name ietf-netconf-acm
   rule-type data-node /nacm/rule-list
   access-operations all
```

Enabling NACM

NACM is disabled on the router by default. Users with root-lr or 'aaa' write task privilege users can enable/disable the NACM via CLI.

To enable NACM, use the following command in the Global configuration mode:

```
Router(config)#aaa authorization nacm default local
```

Cisco IOS XR Software Release 7.4.1 introduces support for external group names.

The external group names are added to the list of local group names to determine the access control rules. External group names are preferred from the list:

```
Router(config)#aaa authorization nacm default prefer-external group tacacs+ local
```

The `local` keyword refers to the locald (AAA local database) and not the NACM database.

Only external group names will be used to determine the access control rules:

```
Router(config)#aaa authorization nacm default only-external local
```

Verification

Use the **show nacm summary** command to verify the default values after enabling NACM:

```
Router# show nacm summary
Mon Jan 15 16:47:43.549 UTC
 NACM SUMMARY
 ------------------------------------------------------------------------------
 Enable Nacm : True
 Enable External Groups : True
 Number of Groups : 0
 Number of Users : 0
 Number of Rules : 0
 Number of Rulelist : 0
 Default Read : permit
 Default Write : deny
 Default Exec : permit
 Denied Operations : 0
 Denied Data Writes : 0
 Denied Notifications : 0
```

Associated Commands

- `Router#show nacm summary`
- `Router#show nacm users [user-name]`
- `Router#show nacm rule-list [rule-list-name] [rule [rule-name]]`
- `Router#show nacm groups [group-name]secret`

Verify the NACM Configurations

Use the **show nacm summary** command to verify the NACM configurations:

```
Router# show nacm summary
Mon Jan 15 17:02:46.696 UTC
 NACM SUMMARY
 ------------------------------------------------------------------------------
 Enable Nacm : True
 Enable External Groups : True
 Number of Groups : 3
 Number of Users : 3
 Number of Rules : 4
 Number of Rulelist : 2
 Default Read : permit
 Default Write : permit
 Default Exec : permit
 Denied Operations : 1
```

```
 Denied Data Writes : 0
 Denied Notifications : 0
-----------------------------------------------------------------------------------
```

Associated Commands

- Router#**show nacm summary**
- Router#**show nacm users [user-name]**
- Router#**show nacm rule-list [rule-list-name] [rule [rule-name]]**
- Router#**show nacm groups [group-name]secret**

Disabling NACM

There are two ways you can disable NACM. Use one of the following commands:

Configuring NACM authorization as none:

```
Router(config)# aaa authorization nacm default none
```

or

Using no form of AAA authorization command:

```
Router(config)# no aaa authorization nacm default
```

Verification

Use the **show nacm summary** command to verify the default values after disabling NACM:

```
Router# show nacm summary

Mon Jan 15 17:02:46.696 UTC
 NACM SUMMARY
 -------------------------------------------------------------------------------
 Enable Nacm : False
 Enable External Groups : True
 Number of Groups : 0
 Number of Users : 0
 Number of Rules : 0
 Number of Rulelist : 0
 Default Read : permit
 Default Write : deny
 Default Exec : permit
 Denied Operations : 0
 Denied Data Writes : 0
 Denied Notifications : 0
```

Dynamic Retrieval of NETCONF Access Control Model Policies

Table 13: Feature History Table

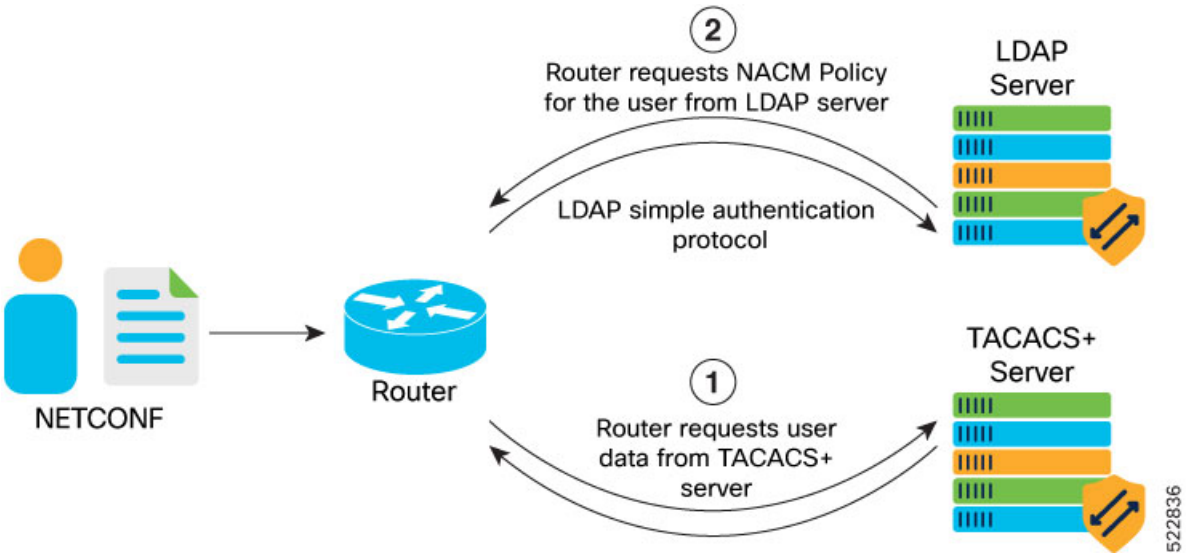
Feature Name	Release Information	Description
Dynamic Retrieval of NETCONF Access Control Model Policies	Release 7.8.1	Your router now retrieves the NETCONF Access Control Model (NACM) policies or rules on-demand for an authorized user from a remote Lightweight Directory Access Protocol (LDAP) server to validate each NETCONF operation. As the policies are stored in an external server and retrieved dynamically, this feature eliminates the need to manually update policies on a per-router basis. Before this release, your router supported static NACM, where the NACM policies or rules were stored locally, requiring manual policy updates on each router. This feature introduces the **nacm enable-external-policies** command.

When you log in to the router using a NETCONF interface, the router authenticates the user credentials, and the authorization check is done for exec service and NETCONF service. After a successful NETCONF service authorization, the user is authorized to perform NETCONF operations or access data nodes within a given RPC based on the rules obtained from the external LDAP server. Unlike in static NACM, where the authorization policies are stored locally, in dynamic NACM, the router retrieves and stores these authorization policies for the authenticated user from the external server dynamically in a secure transfer manner. These policies are used to authorize the NETCONF operations.

LDAP server stores NACM policies. You must configure the LDAP server with the policies (NACM rule-list and rules) for the user or the user group.

The TACACS+ servers contain the netconf service configuration that contains group-mapping, and information to query LDAP server for retrieving the NACM policies.

Figure 2: Workflow of Dynamic NACM



The work flow of the Dynamic NACM is as follows:

1. Router requests the following information from the TACACS+ server:
   - User and user groups
   - LDAP server contact
   - Home directory, and so on

   For a successful authorization, the TACACS+ server responds with `nacm-groups`, `basedn`, `filter`, `map`, and `timestamp` as attribute-value pairs.

   If TACACS+ server becomes unreachable, authorizations of the NETCONF operations use locally defined NACM policies.

2. Router requests the following information from the LDAP server:
   - User's NACM policy

   LDAP server responds with user NCAM policies.

   The authorization policies obtained for a given authenticated user are internally committed to running configuration on the router. If the retrieved policies not required, such policies have to be deleted from the running configuration.

When the router receives a NETCONF service authorization response having a new timestamp attribute-value pair as compared to the timestamp of the policy that is existing on the router, a dynamic policy is downloaded from the LDAP server. The dynamic policies are stored (cached) in the static NACM database.

Configure Dynamic NACM

Configuring dynamic NACM involves the following tasks.

- Router Configuration

- Configure Router-to-LDAP Server Communication, on page 115
- Configure TACACS+ Server Profile, on page 115
- Configure LDAP Server Profile, on page 116
- Enable Dynamic NACM, on page 116

- TACACS+ Server Configuration
- LDAP Server Configuration

Router Configuration

This section provides router configuration for dynamic NACM, which includes establishing a communication between router and LDAP server, LDAP and TACACS+ server profile configurations on router.

Configuring a router for dynamic NACM involves the following tasks:

- Configure Router-to-LDAP Server Communication
- Configure TACACS+ Server Profile
- Configure LDAP Server Profile
- Enable Dynamic NACM

Configure Router-to-LDAP Server Communication

LDAP communication is established between LDAP client running on router and LDAP server, using simple authentication protocol. Use LDAP server host configuration on router to communicate with LDAP server.

For configuration procedure, see Configure LDAP Server Profile, on page 116.

You can use `Cisco-IOS-XR-aaa-ldapd-cfg.yang` file to configure LDAP parameters such as `connect-timeout`, `bind-distinguished-name`, and `bind-password` values for the LDAP and router connectivity.

Configure TACACS+ Server Profile

The TACACS+ client sends the NETCONF authorization request to LDAP server to retrieve nacm_group and LDAP url attributes.

Configuration Example

```
Router# configure
Router(config)# tacacs-server host 10.105.236.101 port 7010
Routers(config-tacacs-host)# key 7 00071A150754
Routers(config-tacacs-host)# commit
```

Running Configuration

```
Router# show run
 tacacs-server host 10.105.236.101 port 7010
  key 7 00071A150754
 !
```

Configure LDAP Server Profile

LDAP communication is established between LDAP client running on router and LDAP server located externally using a simple authentication protocol. Use **ldap-server host** command to configure the LDAP server host (ldap-server) to communicate with LDAP server through CLI.

Note You can configure only one LDAP server host.

Table 14: LDAP Server Host Configuration Parameters

Attribute	Description
ip-address	LDAP server IP address. This is mandatory.
port-number	The port number to connect to the LDAP server. The default value is 389 (LDAP) or 636 (LDAPS). The port value ranges between 1- 65,535.
bind-dn	The Distinguished Name (DN) to bind to the LDAP server. This is mandatory for Authentication.
bind-password	The password to use to bind to the LDAP server. This is mandatory for authentication.
Connect-timeout	Connection establishment time-out between LDAP client and LDAP server. The value ranges between 1–1000 seconds. Default time is five seconds. You can perform three attempts upon bind timeout. If the bind does not respond within three attempts, the server is marked as Dead and router connects to the next available server which is marked as UP.

Configuration Example

```
Router# configure
Router(config)# ldap-server host 10.105.236.10
Router(config-ldap-host)# bind-dn cn=admin,dc=cisco,dc=com
Router(config-ldap-host)# bind-password lablab
Router(config-ldap-host)# connect-timeout 10
Router(config-ldap-host)# commit
```

Running Configuration

```
Router# sh run ldap-server host
 ldap-server host 10.105.236.10 port 389
 bind-dn cn=admin,dc=cisco,dc=com
 bind-password 7 04570A0403204E
 connect-timeout 10
!
```

Enable Dynamic NACM

You can configure NACM either through NETCONF client or CLI.

Note The dynamic policies once configured are not removed. To remove these policies, unconfigure those policies from the running configuration

Configuration Example

To enable dynamic NACM, use the following command in the global configuration mode:

```
Router(config)# nacm enable-external-policies
```

TACACS+ Server Configuration

This section provides TACACS+ server configuration for dynamic NACM, with a set of newly introduced attribute-value pairs.

Cisco IOS XR software Release 7.8.1 introduces **BaseDN**, **filter**, **map**, and **timestamp** attribute-value pairs with which TACACS+ server is configured in the user profile.

Table 15: Attribute-value pair of TACACS+ Server

Attribute-value pair	Description
BaseDN	LDAP client (aaa_ldapd) uses base distinguished name (baseDN) to search for the NACM policy in the LDAP server.
filter	The LDAP filter in the search operation to determine the existence of a specific attribute or an object.
map	Customized name for nacmRuleList.
timestamp	The time at which the NACM policy for the group has changed at the LDAP server.

Configuration Example

The following configuration shows the TACAC+ configuration with LDAP attributes.

```
user = netconf_user1 {
default service = permit
global = cleartext lab
opap = cleartext "lab"
member = aaa-india

service = exec {
   task = "#root-lr,#cisco-support"
   idletime = 2
}
service = netconf {
   nacm-group = “FULL-ACCESSGROUP”
   basedn = “nacmRuleList=FULL-ACCESS,gtacdomain=IPNSG,dc=domain,dc=gtac,dc=cisco,dc=net”
     filter = “(|(objectclass=nacmRuleList)(objectclass=nacmRule))”
     map nacmRuleList profile
  timestamp = 1638169449
 }
}
```

LDAP Server Configuration

This section provides the schema and rule-lists configuration on the LDAP server for dynamic NACM.

LDAP schema rules and rule-lists must be defined in similar way as defined in the `NACM RFC 8341` YANG model.

Schema

A sample LDAP schema for dynamic NACM is as followed:

```
olcAttributeTypes: {0}( 1.3.6.1.4.1.1234.101 NAME 'nacmRuleName' DESC ' Name of the rule'
EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32} SINGLE-VALUE )
olcAttributeTypes: {1}( 1.3.6.1.4.1.1234.102 NAME 'nacmRuleIndex' DESC  'Order of the rule'
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )
olcAttributeTypes: {2}( 1.3.6.1.4.1.1234.105 NAME 'nacmModuleName' DESC  'Name of the YANG
 module associated with this rule' EQUALITY caseIgnoreMatch SYNTAX
1.3.6.1.4.1.1466.115.121.1.15{32} SINGLE-VALUE )
olcAttributeTypes: {3}( 1.3.6.1.4.1.1234.106 NAME 'nacmRuleType' DESC 'Choice between 1=
rpc, 2=data-node or 3=notification' SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )
olcAttributeTypes: {4}( 1.3.6.1.4.1.1234.107 NAME 'nacmRuleData' DESC 'XPath
instance-identifier associated with the data node controlled by this rule or rpc-name or
notification-name' EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32}
SINGLE-VALUE )
olcAttributeTypes: {5}( 1.3.6.1.4.1.1234.109 NAME 'nacmAccessOperations ' DESC 'Access
operations associated with this rule. CRUDX bits (Create-Read-Update-Delete-eXecute-ALL)
value' SYNTAX 1.3.6.1.4.1.1466.115.121.1.6 SIN GLE-VALUE )
olcAttributeTypes: {6}( 1.3.6.1.4.1.1234.110 NAME 'nacmAction' DESC 'Action taken by the
server when a particular rule matches' SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )
olcAttributeTypes: {7}( 1.3.6.1.4.1.1234.113 NAME 'nacmRule' DESC 'NACM Rule' EQUALITY
caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE -VALUE )
olcAttributeTypes: {8}( 1.3.6.1.4.1.1234.103 NAME 'nacmRuleListName' DESC 'Name of the
rulelist' EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466. 115.121.1.15{32} SINGLE-VALUE
)
olcAttributeTypes: {9}( 1.3.6.1.4.1.1234.104 NAME 'nacmRuleListIndex' D ESC 'Order of the
rulelist' SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VAL UE )
olcAttributeTypes: {10}( 1.3.6.1.4.1.1234.135 NAME 'nacmRuleListGroup' DESC 'NACM Group
that will be assigned the associated access defined by the  nacmRuleList' EQUALITY
caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.1 5{32} SINGLE-VALUE )
olcAttributeTypes: {11}( 1.3.6.1.4.1.1234.111 NAME 'nacmLastModifiedTim e' DESC 'date/time
 the ruleList was last modified' EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
 SINGLE-VALUE )
olcAttributeTypes: {12}( 1.3.6.1.4.1.1234.112 NAME 'nacmRuleList' DESC 'NACM set of Rules'
 EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
olcAttributeTypes: {13}( 1.3.6.1.4.1.1234.114 NAME 'nacmNACMGlobal' DESC 'Global NACM
settings' EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.1 15.121.1.15 SINGLE-VALUE )
olcAttributeTypes: {14}( 1.3.6.1.4.1.1234.120 NAME 'nacmEnableNACM' DESC 'Boolean enable
or disable NACM on device' SYNTAX 1.3.6.1.4.1.1466.115.12 1.1.7 SINGLE-VALUE )
olcAttributeTypes: {15}( 1.3.6.1.4.1.1234.121 NAME 'nacmReadDefault' DE SC 'Read Access
default' SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )
olcAttributeTypes: {16}( 1.3.6.1.4.1.1234.122 NAME 'nacmWriteDefault' DESC 'Write Access
default' SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )
olcAttributeTypes: {17}( 1.3.6.1.4.1.1234.123 NAME 'nacmExecDefault' DESC 'Exec Access
default' SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )
olcAttributeTypes: {18}( 1.3.6.1.4.1.1234.124 NAME 'nacmEnableExternalG roups' DESC 'Use
external groups' SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGL E-VALUE )
olcAttributeTypes: {19}( 1.3.6.1.4.1.1234.115 NAME 'nacmNACMGroup' DESC 'NACM Group' EQUALITY
 caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15  SINGLE-VALUE )
olcAttributeTypes: {20}( 1.3.6.1.4.1.1234.130 NAME 'nacmGroupName' DESC 'NACM Group Name'
EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121 .1.15{32} )
olcAttributeTypes: {21}( 1.3.6.1.4.1.1234.131 NAME 'nacmUsersNACM' DESC  'List of users'
EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32} )
olcObjectClasses: {0}( 1.3.6.1.4.1.1235.100 NAME 'nacmRuleList' DESC 'NACM set of Rules OC'
```

```
 SUP top STRUCTURAL MUST ( nacmRuleList $ nacmRuleListName $ nacmRuleListGroup ) MAY (
nacmRuleListIndex $ nacmLastModifiedTime $ description ) )
olcObjectClasses: {1}( 1.3.6.1.4.1.1235.110 NAME 'nacmRule' DESC 'NACM Rule OC' SUP top
STRUCTURAL MUST ( nacmRule $ nacmRuleName $ nacmModuleName $ nacmAccessOperations $ nacmAction
 $ nacmRuleList ) MAY ( nacmRuleIndex $ description $ nacmRuleType $ nacmRuleData ) )
olcObjectClasses: {2}( 1.3.6.1.4.1.1235.120 NAME 'nacmNACMGlobal' DESC 'Global NACM settings
 OC' SUP top STRUCTURAL MAY ( nacmNACMGlobal $ nacmEna bleNACM $ nacmReadDefault $
nacmWriteDefault $ nacmExecDefault $ nacmEnable ExternalGroups $ nacmLastModifiedTime ) )
olcObjectClasses: {3}( 1.3.6.1.4.1.1235.130 NAME 'nacmNACMGroup' DESC ' NACM Group OC' SUP
 top STRUCTURAL MUST ( nacmGroupName $ nacmUsersNACM ) MAY nacmNACMGroup )
```

Note You can use `olcAttributeTypes` and `olcObjectClasses` as per your setup and requirement.

Rule-lists

The LDAP database must be updated with the user NACM policies.

Configuration

Use `show running-config` command to view the LDAP server configuration on the router.

```
Router# show running-config
 nacm rule-list 202 Netconf-READONLY
  rule 1 rule1
   action permit
   module-name *
   access-operations read
  !
  groupnames READONLYGROUP
 !
 nacm rule-list 201 Netconf-FULL-ACCESS
  rule 1 rule1
   action permit
   module-name *
   access-operations all
  !
  groupnames FULL-ACCESSGROUP
 !
```

Note Dynamically downloaded rule-lists are indexed from 201.

Configure an LDAP source interface

Starting with Cisco IOS XR Release 26.3.1, specify the source interface for LDAP packets.

You can optionally specify the VRF for the source-interface configuration.

Procedure

Configure the source interface for LDAP packets.

Example:

```
RP/0/RP0/CPU0:ios(config)# ldap-server source-interface <interface> [vrf <vrf-name>]
```

Use this command syntax:

- Specify an interface that the platform supports. For example, `MgmtEth`.
- Use the interface name in Rack/Slot/Instance/Port (`R/S/I/P`) format when applicable. For example, `MgmtEth 0/RP0/CPU0/0`.
- The `vrf` keyword is optional. Omit it to configure only the source interface.

LDAP traffic is bound to the specified interface and, when configured, the specified VRF.

Dynamic NACM using LDAP over TLS Authentication

Table 16: Feature History Table

Feature Name	Release Information	Description
Securely retrieve NACM policies using LDAP over TLS connection	Release 7.9.1	You can now securely retrieve the NETCONF Access Control Model (NACM) policies or rules from a remote Lightweight Directory Access Protocol (LDAP) server using Transport Layer Security (TLS) authentication. With TLS authentication, the communication between the router and the LDAP server is encrypted for security. Before this release, the communication between the LDAP server and the router was not secured.

You can use the LDAP over TLS (LDAPS) communication to request and retrieve information from remote LDAP server in a secure manner. A maximum of 11 LDAP servers are supported.

The following procedure shows the steps involved in generating the Certification Authority (CA) certificate, adding the CA certificate to the trustpoint and configuring the LDAP server and router to download the NACM policies.

Before you begin

Setup TACACS server. For more information, see Configuring TACACS+ Server Groups, on page 74.

Procedure

Step 1 Add or update the configuration file on the TACACS server as shown in the following example:

Example:

```
user = nacm_user4 {
     default service = permit
     global = cleartext lab
     opap = cleartext "lab"
     member = aaa-member
     service = exec {
```

```
            task = "#serviceadmin"
            idletime = 2
        }
        service = netconf {
           nacm-groups = "READONLY-ACCESSGROUP"
           basedn = "nacmRuleList=Netconf-READONLY-ACCESS,cn=LEAF-XR,ou=users,dc=cisco,dc=com"
           filter = "(|(objectclass=nacmRule)(objectclass=nacmRuleList))"
           map = "nacmRuleList profile"
           timestamp = 1638169449
        }
}
```

Step 2 Enable NACM authorization.

Example:

```
Router(config)#aaa authorization nacm default group tacacs+ local
```

Step 3 Configure the LDAP server.

Example:

```
Router(config)#ldap-server host 172.27.74.235 port 636
Router(config-ldap-host)#bind-dn “”
Router(config-ldap-host)#bind-password “”
```

The bind-dn and bind-password commands accept input values. If certificate authentication is used, the value is null ("").

Step 4 Configure the parameters for TLS communication.

a) Generate RSA key pair for the router. The RSA keys are generated in pairs—a public RSA key and a private RSA key. If the router already has RSA keys when you issue this command, a message is displayed to replace the existing keys with new keys. The keys are generated and saved in the secure NVRAM.

Example:

```
Router#crypto key generate rsa crl
Wed Mar 29 14:13:19.368 UTC
The name for the keys will be: crl
  Choose the size of the key modulus in the range of 512 to 4096 for your General Purpose Keypair.

  Choosing a key modulus greater than 512 may take a few minutes.

How many bits in the modulus [2048]:
Generating RSA keys ...
Done w/ crypto generate keypair
[OK]
```

b) Check that the key pair is generated successfully.

Example:

```
Router#show crypto key mypubkey rsa
Wed Mar 29 14:13:44.592 UTC
Key label: crl
Type     : RSA General purpose
Size     : 2048
Created  : 14:13:25 UTC Wed Mar 29 2023
Data     :
 30820122 300D0609 2A864886 F70D0101 01050003 82010F00 3082010A 02820101
 00A6490A D2184AE0 78F0D4C7 3491886D 6ED679DE 31833CBF B1D0CFA9 33112169
 FDC3443B 79C478D3 B8CC05FB 9810D2E4 E3782733 BFCA7CDD EE56CE5B C98ADF57
 C0D9DE72 D4915A2A 298313D8 A17ABA48 6FA199CE F661F26B 608130B0 F08363DE
 0BC2DDCE 2B79ADA2 D23C9905 96380FEA 60DA6AE8 A38DDEA4 F2233532 2B0788BF
 80BC734B 6CD585D1 60519EFF C65363D2 C98CA384 878F7078 6AE68C81 BE59C09B
 EAC211A9 49D4C04A 3187EF8E 8AA357F7 754F1B9E 80276462 7DC249BF 2649BCD3
```

```
B6C2F6F0 A41926A5 7297F7D9 F3403928 194102F7 601E4CE4 A7190F8F CE8DBE24
082C3D7A 24CA8C1C 2323C7F7 499C1BD6 21DD218C F1F72740 978AB9F4 801FB38B
09020301 0001
```

c) Configure a trustpoint with the server so that the router can verify the certificates issued to peers.

Example:

```
Router#configure
Router(config)#crypto ca trustpoint ldaps
Router(config-trustp)#subject-name
C=IN,ST=Karnataka,L=Bengaluru,O=cisco,OU=department,CN=client.cisco.com
Router(config-trustp)#enrollment url terminal
Router(config-trustp)#enrollment retry count 99
Router(config-trustp)#enrollment retry period 1
Router(config-trustp)#rsakeypair crl
Router(config-trustp)#domain name cisco.com
```

The retry count is the number of times the router resends a certificate request when the router does not receive a certificate from the previous request. The range is from 1 to 100. If no retry count is specified, the default value is 10. The retry period is the time between certificate requests issued to a certification authority (CA) from the router. The range is from 1 to 60 minutes. The default is 1 minute.

d) Authenticate the CA. Configure the security public key infrastructure (PKI) trace options.

Example:

Router:

```
Router#crypto ca authenticate ldaps
Mon Mar 20 02:20:18.044 UTC

Enter the base 64 encoded certificate.
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
MIIF6TCCA9GgAwIBAgIJAK7Auq53lyF3MA0GCSqGSIb3DQEBCwUAMIGKMQswCQYD
VQQGEwJJTjEMMAoGA1UECAwDS2FyMQ4wDAYDVQQHDAVCbG9yZTEOMAwGA1UECgwF
------------ Certificate details truncated for brevity ------------
hUFUx56f158KIiDx4SgwLZL4+UXLM+wxbpSgB9sQVz3f1yLhMuf9KgHZjE6O2Rr3
5te9emSo64ros6M01sQ5rWsPdqYC/jlN3M7eBIw=
-----END CERTIFICATE-----

  Serial Number  : AE:00:BA:AE:77:77:21:97
  Subject:

emailAddress=user@cisco.com,CN=server.example.com,OU=Test,O=Cisco,L=Bengaluru,ST=Karnataka,C=India

  Issued By      :

emailAddress=user@cisco.com,CN=server.example.com,OU=Test,O=Cisco,L=Bengaluru,ST=Karnataka,C=India

  Validity Start : 17:07:28 UTC Mon Mar 20 2023
  Validity End   : 17:07:28 UTC Tue Mar 19 2024
  SHA1 Fingerprint:
            C500X79A7A7FBBB668D009554BDA80698DABC6A4
Do you accept this certificate? [yes/no]: yes
```

The router authenticates the CA by obtaining the self-signed certificate that contains the public key.

e) Enroll the device certificate with CA.

Example:

Router:

```
Router#crypto ca enroll ldaps
Mon Mar 20 02:24:05.270 UTC
% Start certificate enrollment ...
% Create a challenge password. You will need to verbally provide this
  password to the CA Administrator in order to revoke your certificate.
% For security reasons your password will not be saved in the configuration.
% Please make a note of it.

Password:
Re-enter Password:

% The subject name in the certificate will include:
C=India,ST=Karnataka,L=Bengaluru,O=Cisco,OU=test,CN=client.cisco.com
% The subject name in the certificate will include: R2.cisco.co
% Include the router serial number in the subject name? [yes/no]: no
% Include an IP address in the subject name? [yes/no]: no
    Fingerprint:  44443744 33377244 45563033 44334668
Display Certificate Request to terminal? [yes/no]: yes
Certificate Request follows:

MIIDJjCCAg4CAQAwgYExCzAJBgNVBAYTAklOMQwwCgYDVQQIDANLYXIxDjAMBgNV
BAcMBUJsb3JlMQ4wDAYDVQQKDAVDaXNjbzENMAsGA1UECwwEdGVzdDEZMBcGA1UE
------------ Certificate details truncated for brevity ------------
ugcy9fUHnV+YoKD3pg3p8Cutg2Tudm1DYj4U8BBbp+YZNMc8BhHX3F8Cx4JOvioR
BKo4IfxPi0HspcQDDDdivNtl6JRJA+8scGHajsVgI8eXE+5PxY7ejsbS

---End - This line not part of the certificate request---
Redisplay enrollment request? [yes/no]:
```

Step 5 Copy the generated CA request certificate to `/etc/openldap/cacerts/ca-req.pem` file with the start and end tags.

Example:

LDAP server terminal:

```
-----BEGIN CERTIFICATE----- <.data from router.> -----END CERTIFICATE-----
```

Step 6 Generate `sys-cert.pem` router certificate.

Example:

```
Serevr>openssl ca -md sha256 -config /etc/pki/tls/openssl.cnf -keyfile /etc/pki/CA/ca.key -cert
/etc/pki/CA/ca.cert.pem -in /etc/openldap/cacerts/ca-req.pem -out /etc/openldap/cacerts/sys-cert.pem
```

The `ca-req.pem` certificate is configured during router configuration. The `ca.cert.pem` key is created during server setup.

Step 7 Import the generated `sys-cert.pem` certificate to the router.

Example:

```
Router#crypto ca import ldaps certificate
```

Step 8 Check that the certificate is imported successfully.

Example:

```
Router#show crypto ca certificates
Mon Mar 20 02:23:35.438 UTC

Trustpoint       : ldaps
==================================================
CA certificate
  Serial Number  : AE:00:BA:AE:77:77:21:97
```

```
  Subject:

emailAddress=user@cisco.com,CN=server.example.com,OU=Test,O=Cisco,L=Bengaluru,ST=Karnataka,C=India
  Issued By      :

emailAddress=user@cisco.com,CN=server.example.com,OU=Test,O=Cisco,L=Bengaluru,ST=Karnataka,C=India
  Validity Start : 17:07:28 UTC Mon Mar 20 2023
  Validity End   : 17:07:28 UTC Tue Mar 19 2024
  SHA1 Fingerprint:
            C500X79A7A7FBBB668D009554BDA80698DABC6A4
```

The certificate details enrolled in trustpoint is displayed.

With this configuration, the LDAPS server is ready for the NETCONF operations to download the NACM rules using TLS communication.

The following example shows the NACM rules added to the LDAP server.

```
# Netconf-READONLY, MTLAB-X-LEAF-PEERING-XR_TL, users, cisco.com
dn: nacmRuleList=Netconf-READONLY,cn=MTLAB-X-LEAF-PEERING-XR_TL,ou=users,dc=example,dc=com
nacmLastModifiedTime: 20220215003
nacmRuleListIndex: 1
nacmRuleListGroup: READONLYGROUP
nacmRuleList: Netconf-READONLY
nacmRuleListName: Netconf-READONLY
objectClass: top
objectClass: nacmRuleList
```

Command Authorization Using Local User Account

Table 17: Feature History Table

Feature Name	Release Information	Feature Description
Command Authorization Using Local User Account	Release 7.5.1	This feature allows locally authenticated users—authenticated by the AAA server internal to the router—to run all XR VM commands even if a remote TACACS+ AAA server is not reachable for authorization. It prevents a complete router lockdown. The feature also prevents remotely authenticated users—authenticated using a remote AAA server (say, TACACS+ server)—from running any non-permitted commands on the router, and thus prevents misuse of user privileges. This feature modifies the **aaa authorization commands default** command to include the **local** option for XR VM command authorization.

Currently, when a user tries to execute a command on XR VM, the router checks to see whether the user has required permissions to execute it. The router does this authorization process in two steps. First, the system compares the task-IDs of the user with the required task-IDs for the command. If the user has all required task-IDs, and if AAA authorization is configured, then the system sends an authorization request to the local or remote AAA server, based on that configuration. Based on the response from the AAA server, the system allows or rejects the command execution. If authorization is not configured or if it configured with option *none*, then the system bypasses authorization check and allows user to execute the command.

Similarly, the existing remote authorization process using TACACS+ server has two options—remote authorization using *tacacs+* and *none*. The authorization process using *TACACS+* option uses an external TACACS+ server for authorization. The authorization using *none* option allows the user to execute the command without any authorization check. TACACS+ authorization has the advantage of fine-tuning authorization rules and providing more control on system access that cannot be otherwise done locally. However, if the remote server is not reachable, a user who leverages TACACS+ authorization might get into an unpredictable state of router, as mentioned in these scenarios:

- Remote authorization using *TACACS+* with failover option as *none* (that is, with the **aaa authorization commands default group tacacs+ none** configuration)

  If TACACS+ server is not reachable, then the system bypasses the authorization check and allows user to execute the command. A user who does not have permission to execute certain commands due to additional authorization rules on the TACACS+ server, then gets permission to execute those commands in this scenario. This action introduces a privilege escalation.

- Remote authorization using *TACACS+* without any failover option (that is, with the **aaa authorization commands default group tacacs+** configuration)

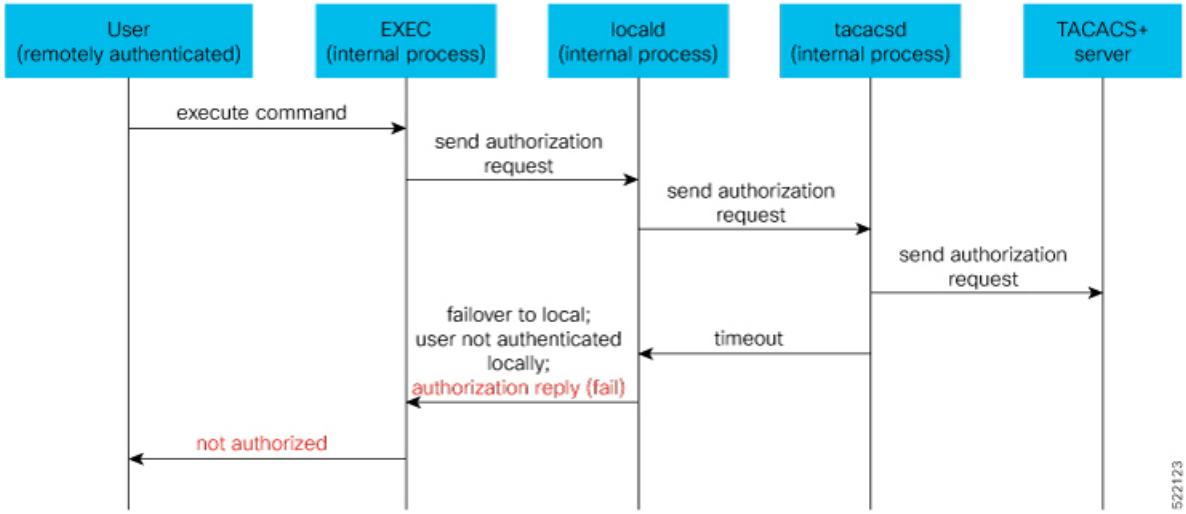
  If TACACS+ server is not reachable, then the system does not authorize the command at all. Because the user then cannot execute any command, the router gets locked out.

With the introduction of command authorization using local user account feature in Cisco IOS XR Software Release 7.5.1, locally authenticated users can execute commands even if a TACACS+ server is not reachable. This behavior is similar to the behavior with the failover option *none*, with the only difference that only locally authenticated users can execute commands in this case. This functionality thereby prevents a complete lockdown of the router as mentioned in one of the previously existing scenarios mentioned earlier. At the same time, the feature also prevents users who are authenticated remotely (that is, TACACS+ authenticated users) from executing any non-permitted command on the router. This behavior in turn helps to prevent any sort of misuse of user privileges on the router.

Call Flow of Command Authorization

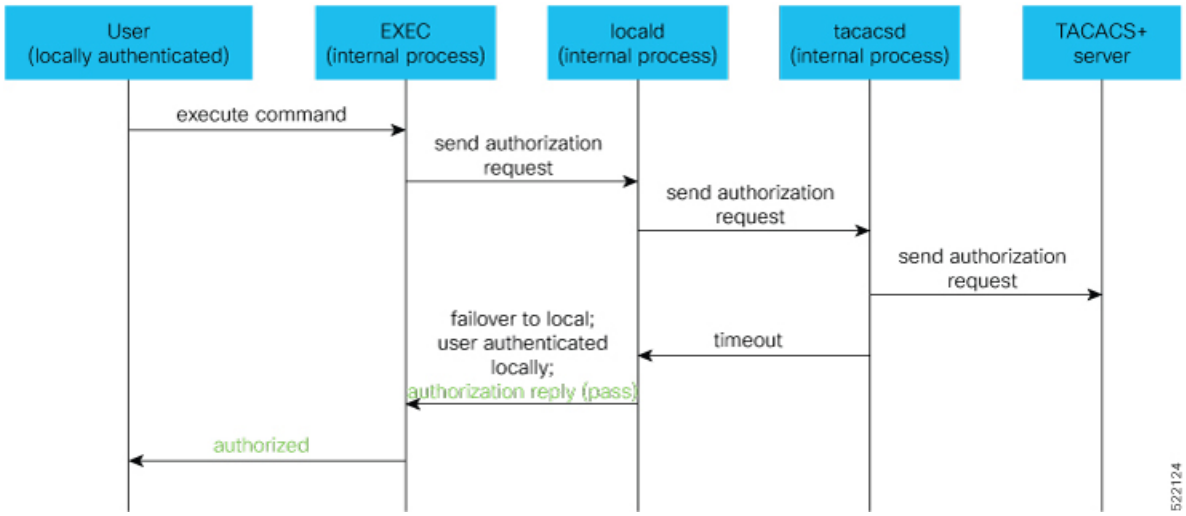
Consider a scenario where the user is remotely authenticated. In the event of timeout from the TACACS+ server, the command authorization fails. The user cannot execute any command until the TACACS+ server is reachable again, thereby preventing misuse of user privileges on the router.

Figure 3: Call Flow of Command Authorization for Remotely Authenticated Users



Consider a scenario where the user is locally authenticated. The command authorization still succeeds even if the authorization request to the TACACS+ server times out. There is no additional check done by the local AAA component in the router. As a result, the user can execute the command irrespective of the fact that the TACACS+ server is not reachable. This functionality prevents a complete lockdown of the router.

Figure 4: Call Flow of Command Authorization for Locally Authenticated Users



Configure Command Authorization Using Local User Account

Guidelines

Although there is no restriction in configuring local command authorization, you must be cautious to prevent any potential lockout due to misconfiguration. For instance, if *local* is the only method of authorization specified for the commands, a remotely authenticated user configuring command authorization using local user account feature cannot execute further commands.

Configuration Example

To configure command authorization using local user account, use the **local** option in the **aaa authorization** command in any of these formats:

```
Router#configure
Router(config)#aaa authorization commands default group tacacs+ local
```

Or

```
Router(config)#aaa authorization commands default local
```

Running Configuration

```
Router#show run aaa
!
aaa authorization commands default group tacacs+ local
!
```

```
Router#show run aaa
!
aaa authorization commands default local
!
```

Verification

```
Router#show user authentication method
local
```

Feature Behavior and Use Case Scenarios

Feature Behavior With Various Local Command Authorization Options

This table lists the feature behavior scenarios with various local command authorization options.

Table 18: Feature Behavior with Various Local Command Authorization Options

AAA Configuration	Expected Behavior
aaa authorization commands default group tacacs+ local	If TACACS+ server is not reachable, system allows locally authenticated users to execute the command. If TACACS+ server is reachable and if it returns an authorization failure, then the system does not perform any failover to local authentication with this configuration.
aaa authorization commands default local	This configuration allows only locally authenticated users to execute commands. System completely blocks remote users from executing any command.
aaa authorization commands default local group tacacs+	In this scenario, system chooses local authorization first and grants access if the user is locally authenticated. If not, the request fails over to TACACS+ server. This combination of command options is useful when both local and remote authenticated users want to execute commands when TACACS+ server is reachable.
aaa authorization commands default local none	Although configurable, this combination of command options does not provide any additional security with respect to user access. It is equivalent to having no authorization.

Use Case Scenarios of Command Authorization

In the following scenarios, local user refers to user whose is authenticated locally and whose profile is available locally, but not available on the remote server (TACACS+ server). Similarly, remote user refers to user whose is authenticated remotely and whose profile is available on the remote server, but not available locally. And, both local user and remote user are considered to have *root-lr* permission to execute the commands, in these scenarios.

Table 19: Use Case Scenarios of Command Authorization

Type of User (local or remote)	AAA Configuration Summary	Use Case Scenario	Expected Behavior
Local and remote user	No command authorization configured	Execute a command	Command authorization succeeds if the required task-IDs are available
Local user	Only *tacacs+ command authorization* configured.	Execute a command when TACACS+ server is reachable	Command authorization fails
		Execute a command when TACACS+ server is not reachable	Command authorization fails

Remote user	Only *tacacs+ command authorization* configured	Execute a command when TACACS+ server is reachable	Command authorization succeeds
			`Router#show run aaa authorization`
			`aaa authorization commands default group tacacs+`

		Execute a command when TACACS+ server is not reachable	Command authorization fails
Local user	Only *tacacs+ command authorization* configured with failover option as *none*.	Execute a command when TACACS+ server is reachable	Command authorization fails

		Execute a command when TACACS+ server is not reachable	Command authorization succeeds
			`Router#show user authentication method`
			`local`

Remote user	Only *tacacs+ command authorization* configured with failover option as *none*.	Execute a command that is restricted only to that user when TACACS+ server is reachable	Command authorization fails
		Execute a command that is restricted only to that user when TACACS+ server is not reachable	Command authorization succeeds

Type of User (local or remote)	AAA Configuration Summary	Use Case Scenario	Expected Behavior

Local user	Only *local command authorization* configured.	Execute a command	Command authorization succeeds
			`Router#show run aaa authentication`
			`aaa authentication login default group tacacs+ local`

Remote user	Only *local command authorization* configured.	Execute a command	Command authorization fails
Local user	Only *tacacs+ command authorization* configured with failover option as *local*.	Execute a command when TACACS+ server is reachable	Command authorization fails

		Execute a command when TACACS+ server is not reachable	Command authorization succeeds
			`Router#show run aaa authorization`
			`aaa authorization commands default group tacacs+ local`

Remote user	Only *tacacs+ command authorization* configured with failover option as *local*.	Execute a command when TACACS+ server is reachable	Command authorization succeeds
			`Router#show run aaa authorization`
			`aaa authorization commands default group tacacs+ local`

		Execute a command when TACACS+ server is not reachable	Command authorization fails

Additional References

The following sections provide references related to configuring AAA services.

Related Documents

Related Topic	Document Title
AAA services commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Authentication, Authorization, and Accounting Commands on* the Cisco ASR 9000 Series Router in the

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

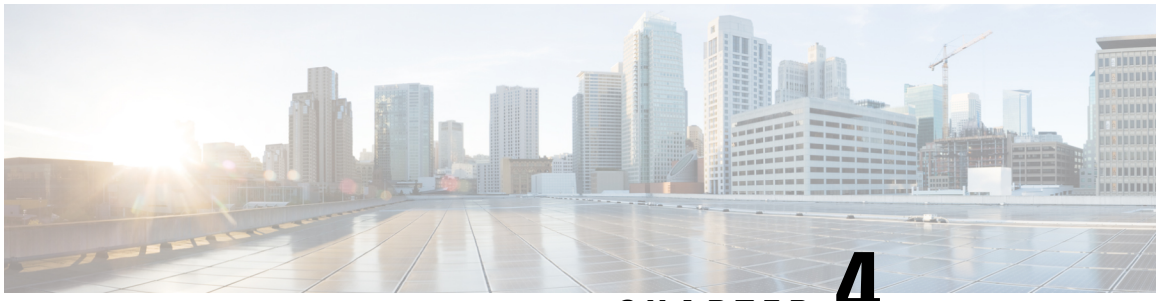
MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	—

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport



CHAPTER 4

Implementing Certification Authority Interoperability

Certification authority (CA) interoperability is provided in support of the IP Security (IPSec), Secure Socket Layer (SSL), and Secure Shell (SSH) protocols. This module describes how to implement CA interoperability.

CA interoperability permits Cisco ASR 9000 Series Router devices and CAs to communicate so that your device can obtain and use digital certificates from the CA. Although IPSec can be implemented in your network without the use of a CA, using a CA provides manageability and scalability for IPSec.

Note IPSec is supported only for Open Shortest Path First version 3 (OSPFv3).

For a complete description of the public key infrastructure (PKI) commands used in this chapter, refer to the *Public Key Infrastructure Commands* module in *System Security Command Reference for Cisco ASR 9000 Series Routers*.

Feature History for Implementing Certification Authority Interoperability

Release	Modification
Release 3.7.2	This feature was introduced.
Release 7.0.1	Added topics on Integrating Cisco IOS XR and Crosswork Trust Insights.
Release 7.3.1	Added support for Ed25519 Public-Key Signature System.
Release 7.3.1	Added support for verifying authenticity of RPM packages using runtime and install time fingerprint.
Release 7.3.1	Added support to collect filesystem inventory.
Release 7.3.1	Added support for optimizations via IMA.

- Prerequisites for Implementing Certification Authority, on page 134
- Restrictions for Implementing Certification Authority, on page 134
- Information About Implementing Certification Authority, on page 135
- How to Implement CA Interoperability, on page 138

• Configuration Examples for Implementing Certification Authority Interoperability, on page 152
• Expiry Notification for PKI Certificate, on page 154
• Automatic renewal of PKI certificate, on page 157
• Integrating Cisco IOS XR and Crosswork Trust Insights, on page 159
• Support for Ed25519 Public-Key Signature System, on page 174
• Where to Go Next, on page 176
• Additional References, on page 176

Prerequisites for Implementing Certification Authority

The following prerequisites are required to implement CA interoperability:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- You need to have a CA available to your network before you configure this interoperability feature. The CA must support Cisco Systems PKI protocol, the simple certificate enrollment protocol (SCEP) (formerly called certificate enrollment protocol [CEP]).

Restrictions for Implementing Certification Authority

- Cisco IOS XR software doesn't support CA server public keys greater than 2048 bits.
- Starting with Cisco IOS XR software version 7.3.3, the server certificates (leaf certificates) in the router must have a Fully Qualified Domain Name (FQDN) in the Common Name (CN) field.
- To add an IP address in the Subject Alternate Name (SAN) field of server certificates, add the extension type as IP address in the certificate. If the IP address extension type configuration isn't available, use the **crypto ca fqdn-check ip-address allow** command for the router to validate the IP address in the SAN field successfully.
- Starting Cisco IOS XR Software Release 7.4.1, we mandate the below X509 certificate Subject Alternate Name (SAN) fields and domain name server configuration to validate SAN. TLS connection cannot be established if there is no domain name-server is configured.

  Here are some key-points regarding SAN field:

  - SAN must be a fully-qualified domain name. For example, DNS:smartreceiver.cisco.com
  - SAN must be a critical extension in the absence of Common Name (CN).
  - If the SAN cannot be represented as a FQDN, then it must be configured with GeneralName field as IP Address but not as DNS. For example, **IP address: 192.0.2.1**

  To configure domain name-server use the **domain name-server** *ip-address*.

  To configure domain name-server with VRF, use the following commands:

  - **domain vrf name-server***ip-address*
  - Use the **crypto ca** *trustpoint-name* **vrf** *vrf-name* command when you are using VRF.

- Use the **crypto ca trustpoint Trustpool vrf** *vrf-name* command for smart-licensing.

For Static Domain Name Configuration, use the **domain ipv4 host** *host-name ip-address* command. and for configuring static domain name using VRF, use the **domain ipv4 vrf** *vrf-name host-name ip-address* command.

- Starting Cisco IOS XR Software Release 7.4.2, you can bypass FQDN and IP address check in SAN by configuring **crypto ca fqdn-check ip-address allow.**

Information About Implementing Certification Authority

To implement CA, you need to understand the following concepts:

Supported Standards for Certification Authority Interoperability

Cisco supports the following standards:

- IPSec—IP Security Protocol. IPSec is a framework of open standards that provides data confidentiality, data integrity, and data authentication between participating peers. IPSec provides these security services at the IP layer; it uses Internet Key Exchange (IKE) to handle negotiation of protocols and algorithms based on local policy, and to generate the encryption and authentication keys to be used by IPSec. IPSec can be used to protect one or more data flows between a pair of hosts, a pair of security gateways, or a security gateway and a host.



Note IPSec is supported only for Open Shortest Path First version 3 (OSPFv3).

- IKE—A hybrid protocol that implements Oakley and Skeme key exchanges inside the Internet Security Association Key Management Protocol (ISAKMP) framework. Although IKE can be used with other protocols, its initial implementation is with the IPSec protocol. IKE provides authentication of the IPSec peers, negotiates IPSec keys, and negotiates IPSec security associations (SAs).
- Public-Key Cryptography Standard #7 (PKCS #7)—A standard from RSA Data Security Inc. used to encrypt and sign certificate enrollment messages.
- Public-Key Cryptography Standard #10 (PKCS #10)—A standard syntax from RSA Data Security Inc. for certificate requests.
- RSA keys—RSA is the public key cryptographic system developed by Ron Rivest, Adi Shamir, and Leonard Adelman. RSA keys come in pairs: one public key and one private key.
- SSL—Secure Socket Layer protocol.
- X.509v3 certificates—Certificate support that allows the IPSec-protected network to scale by providing the equivalent of a digital ID card to each device. When two devices want to communicate, they exchange digital certificates to prove their identity (thus removing the need to manually exchange public keys with each peer or specify a shared key at each peer). These certificates are obtained from a CA. X.509 as part of the X.500 standard of the ITU.

Certification Authorities

The following sections provide background information about CAs:

Purpose of CAs

CAs are responsible for managing certificate requests and issuing certificates to participating IPSec network devices. These services provide centralized key management for the participating devices.

CAs simplify the administration of IPSec network devices. You can use a CA with a network containing multiple IPSec-compliant devices, such as routers.

Digital signatures, enabled by public key cryptography, provide a means of digitally authenticating devices and individual users. In public key cryptography, such as the RSA encryption system, each user has a key pair containing both a public and a private key. The keys act as complements, and anything encrypted with one of the keys can be decrypted with the other. In simple terms, a signature is formed when data is encrypted with a user's private key. The receiver verifies the signature by decrypting the message with the sender's public key. The fact that the message could be decrypted using the sender's public key indicates that the holder of the private key, the sender, must have created the message. This process relies on the receiver's having a copy of the sender's public key and knowing with a high degree of certainty that it does belong to the sender and not to someone pretending to be the sender.

Digital certificates provide the link. A digital certificate contains information to identify a user or device, such as the name, serial number, company, department, or IP address. It also contains a copy of the entity's public key. The certificate is itself signed by a CA, a third party that is explicitly trusted by the receiver to validate identities and to create digital certificates.

To validate the signature of the CA, the receiver must first know the CA's public key. Normally, this process is handled out-of-band or through an operation done at installation. For instance, most web browsers are configured with the public keys of several CAs by default. IKE, an essential component of IPSec, can use digital signatures to authenticate peer devices for scalability before setting up SAs.

Without digital signatures, a user must manually exchange either public keys or secrets between each pair of devices that use IPSec to protect communication between them. Without certificates, every new device added to the network requires a configuration change on every other device with which it communicates securely. With digital certificates, each device is enrolled with a CA. When two devices want to communicate, they exchange certificates and digitally sign data to authenticate each other. When a new device is added to the network, a user simply enrolls that device with a CA, and none of the other devices needs modification. When the new device attempts an IPSec connection, certificates are automatically exchanged and the device can be authenticated.

IPSec Without CAs

Without a CA, if you want to enable IPSec services (such as encryption) between two Cisco routers, you must first ensure that each router has the key of the other router (such as an RSA public key or a shared key). This requirement means that you must manually perform one of the following operations:

- At each router, enter the RSA public key of the other router.
- At each router, specify a shared key to be used by both routers.

If you have multiple Cisco routers in a mesh topology and want to exchange IPSec traffic passing among all of those routers, you must first configure shared keys or RSA public keys among all of those routers.

Every time a new router is added to the IPSec network, you must configure keys between the new router and each of the existing routers.

Consequently, the more devices there are that require IPSec services, the more involved the key administration becomes. This approach does not scale well for larger, more complex encrypting networks.

IPSec with CAs

With a CA, you need not configure keys between all the encrypting routers. Instead, you individually enroll each participating router with the CA, requesting a certificate for the router. When this enrollment has been accomplished, each participating router can dynamically authenticate all the other participating routers.

To add a new IPSec router to the network, you need only configure that new router to request a certificate from the CA, instead of making multiple key configurations with all the other existing IPSec routers.

IPSec with Multiple Trustpoint CAs

With multiple trustpoint CAs, you no longer have to enroll a router with the CA that issued a certificate to a peer. Instead, you configure a router with multiple CAs that it trusts. Thus, a router can use a configured CA (a trusted root) to verify certificates offered by a peer that were not issued by the same CA defined in the identity of the router.

Configuring multiple CAs allows two or more routers enrolled under different domains (different CAs) to verify the identity of each other when using IKE to set up IPSec tunnels.

Through SCEP, each router is configured with a CA (the enrollment CA). The CA issues a certificate to the router that is signed with the private key of the CA. To verify the certificates of peers in the same domain, the router is also configured with the root certificate of the enrollment CA.

To verify the certificate of a peer from a different domain, the root certificate of the enrollment CA in the domain of the peer must be configured securely in the router.

During IKE phase one signature verification, the initiator will send the responder a list of its CA certificates. The responder should send the certificate issued by one of the CAs in the list. If the certificate is verified, the router saves the public key contained in the certificate on its public key ring.

With multiple root CAs, Virtual Private Network (VPN) users can establish trust in one domain and easily and securely distribute it to other domains. Thus, the required private communication channel between entities authenticated under different domains can occur.

How IPSec Devices Use CA Certificates

When two IPSec routers want to exchange IPSec-protected traffic passing between them, they must first authenticate each other—otherwise, IPSec protection cannot occur. The authentication is done with IKE.

Without a CA, a router authenticates itself to the remote router using either RSA-encrypted nonces or preshared keys. Both methods require keys to have been previously configured between the two routers.

With a CA, a router authenticates itself to the remote router by sending a certificate to the remote router and performing some public key cryptography. Each router must send its own unique certificate that was issued and validated by the CA. This process works because the certificate of each router encapsulates the public key of the router, each certificate is authenticated by the CA, and all participating routers recognize the CA as an authenticating authority. This scheme is called IKE with an RSA signature.

Your router can continue sending its own certificate for multiple IPSec sessions and to multiple IPSec peers until the certificate expires. When its certificate expires, the router administrator must obtain a new one from the CA.

When your router receives a certificate from a peer from another domain (with a different CA), the certificate revocation list (CRL) downloaded from the CA of the router does not include certificate information about the peer. Therefore, you should check the CRL published by the configured trustpoint with the Lightweight Directory Access Protocol (LDAP) URL to ensure that the certificate of the peer has not been revoked.

To query the CRL published by the configured trustpoint with the LDAP URL, use the **query url** command in trustpoint configuration mode.

CA Registration Authorities

Some CAs have a registration authority (RA) as part of their implementation. An RA is essentially a server that acts as a proxy for the CA so that CA functions can continue when the CA is offline.

How to Implement CA Interoperability

This section contains the following procedures:

Configuring a Router Hostname and IP Domain Name

This task configures a router hostname and IP domain name.

You must configure the hostname and IP domain name of the router if they have not already been configured. The hostname and IP domain name are required because the router assigns a fully qualified domain name (FQDN) to the keys and certificates used by IPSec, and the FQDN is based on the hostname and IP domain name you assign to the router. For example, a certificate named router20.example.com is based on a router hostname of router20 and a router IP domain name of example.com.

SUMMARY STEPS

1. **configure**
2. **hostname** *name*
3. **domain name domain-name**
4. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**hostname** *name* **Example:** `RP/0/RSP0/CPU0:router(config)# hostname myhost`	Configures the hostname of the router.

	Command or Action	Purpose
Step 3	**domain name domain-name** **Example:** `RP/0/RSP0/CPU0:router(config)# domain name mydomain.com`	Configures the IP domain name of the router.
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

RSA key pairs

RSA key pairs play a crucial role in securing IKE key management messages by enabling both signing and encryption. Their use is required before a certificate can be obtained for your router. Specifically, RSA key pairs are used to

- digitally sign IKE key management messages, ensuring their authenticity and integrity
- encrypt IKE key management messages, protecting sensitive information from unauthorized access, and
- satisfy prerequisites for obtaining a digital certificate for the router, which further enhances security in network communications.

Table 20: Feature History Table

Feature Name	Release Information	Feature Description
Syslog warnings for RSA keys and DSA keys	Release 26.1.1	This enhancement ensures Cisco IOS XR devices comply with evolving security standards by sending syslog warnings if weak SSH host keys are found during system reboot where an Rivest, Shamir, and Adelman (RSA) key less than 3072 bits or any Digital Signature Algorithm (DSA) keys are detected. The default RSA key size has been increased from 2048 to 3072 bits to strengthen security. Additionally, DSA keys are no longer auto-generated at boot, and if found, a syslog warning prompts their removal to align with current best practices. This feature introduces these changes: CLI: The **crypto key generate rsa** and **show crypto key mypubkey rsa** commands have been modified.

Syslog warnings for RSA keys and DSA keys

Starting with Cisco IOS XR Release 26.1.1, the default RSA key size is 3072 bits; syslog warning is triggered during system boot for weak SSH host key (RSA key < 3072 bits). Additionally, DSA keys are no longer auto-generated at boot, and if found, a syslog warning prompts their removal. These changes ensure compliance with current cryptographic security standards and enhance device protection against emerging threats.

Generate an RSA key pair

This task generates an RSA key pair.

Note From Cisco IOS XR Software Release 7.0.1 and later, the crypto keys are auto-generated at the time of router boot up. Hence, step 1 is required to be configured only if the RSA host-key pair is not present in the router under some scenarios.

RSA key pairs are used to sign and encrypt IKE key management messages and are required before you can obtain a certificate for your router.

Procedure

Step 1 Execute the **crypto key generate rsa general-keys** command to generate RSA key pairs.

Example:

```
RP/0/RSP0/CPU0:router# crypto key generate rsa general-keys
```

Use the **general-keys** keyword to specify general purpose RSA keys.

Step 2 Execute the **crypto key zeroize rsa** [*keypair-label*] to delete all RSAs from the router.

Example:

```
RP/0/RSP0/CPU0:router# crypto key zeroize rsa key1
```

- Under certain circumstances, you may want to delete all RSA keys from you router. For example, if you believe the RSA keys were compromised in some way and should no longer be used, you should delete the keys.
- To remove a specific RSA key pair, use the *keypair-label* argument.

Step 3 (Optional) Execute the **show crypto key mypubkey rsa** command to display the RSA public keys for your router.

Example:

```
Router# show crypto key mypubkey rsa
Fri Mar 27 14:00:20.954 IST
Key label: the_default
Type : RSA General purpose
Size : 3072
Created : 01:13:10 IST Thu Feb 06 2020
Data :
 30820122 300D0609 2A864886 F70D0101 01050003 82010F00 3082010A 02820101
 00A93DE0 1E485EE3 0E7F0964 C48361D1 B6014BE7 A303D8D6 F7790E92 88E69C4B
 B97B7A9C D1B277E3 1569093C 82BD3258 7F67FB49 94860ECD 34498F1F 59B45757
 F32C8E8F 7CEE23EC C36A43D1 9F85C0D9 B96A14DD DD3BBD4C A1FB0888 EED210A7
 39D9A403 7ACE0F6E 39107226 CA621AD8 6E8102CA 9761B86F D33F2871 9DD16559
 AFCB4729 EFCEDBAF 83DF76E4 9A439844 EE3B1180 4022F575 99E11A2C E25BB23D
 9DD74C81 4E5C1345 D9E3CC79 1B98B1AA 6C06F004 22B901EC 36C099FE 10DE2622
 EB7CE618 9A555769 12D94C90 D9BEE5EA A664E7F6 4DF8D8D4 FE7EAB07 1EF4FEAB
 22D9E55F 62BA66A0 72153CEC 81F2639F B5F2B5C5 25E10364 19387C6B E8DB8990
 11020301 0001
```

Importing a Public Key to the Router

This task imports a public key to the router.

A public key is imported to the router to authenticate the user.

SUMMARY STEPS

1. **crypto key import authentication rsa** [**usage keys** | **general-keys**] [**keypair-label**]
2. show crypto key mypubkey rsa

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**crypto key import authentication rsa** [**usage keys** | **general-keys**] [**keypair-label**]	Generates RSA key pairs.

	Command or Action	Purpose
	Example: `RP/0/RSP0/CPU0:router# crypto key import authentication rsa general-keys`	• Use the **usage keys** keyword to specify special usage keys; use the **general-keys** keyword to specify general-purpose RSA keys. • The *keypair-label* argument is the RSA key pair label that names the RSA key pairs.
Step 2	show crypto key mypubkey rsa **Example:** `RP/0/RSP0/CPU0:router# show crypto key mypubkey rsa`	(Optional) Displays the RSA public keys for your router.

System logs and error messages

These are the system logs and error messages reported when generating key-pairs.

- The router sends these system logs on successful creation of key-pairs:

```
cepki[287]: %SECURITY-CEPKI-6-KEY_INFO : crypto key DSA generated, label:the_default,
modBits:1024
cepki[287]: %SECURITY-CEPKI-6-KEY_INFO : crypto key ECDSA_NISTP256 generated,
label:the_default, modBits:256
```

- The router sends these system logs on deletion of key-pairs:

```
cepki[287]: %SECURITY-CEPKI-6-KEY_INFO : crypto key RSA zeroized, label:user1
cepki[287]: %SECURITY-CEPKI-6-KEY_INFO : crypto key DSA zeroized, label:the_default
```

- The router sends these error messages if you try to overwrite the key-pairs generated in XR Config Mode from XR EXEC mode:

```
Router#conf t
Router(config)#crypto key generate ed25519
Router(config)#commit
Router#crypto key generate ed25519
Cannot execute the command : Operation not permitted
ce_cmd[68727]: %SECURITY-CEPKI-6-ERR_2 : Cannot      execute the command :   Operation
 not permitted
ce_cmd[68736]: %SECURITY-CEPKI-6-ERR : Key is added as part of config mode, key deletion
 is not allowed , delete key from config mode
```

- The router sends these error messages if you try to delete key-pairs generated in XR Config Mode from XR EXEC mode:

```
Router#conf t
Router(config)#crypto key generate ed25519
Router(config)#commit
Router#crypto key zeroize ed25519
Cannot execute the command : Operation not permitted
ce_cmd[68736]: %SECURITY-CEPKI-6-ERR_2 : Cannot execute the command :   Operation not
permitted
```

- The router sends a syslog warning if the DSA keys and weak RSA keys are found during the system boot or process restart.

```
Router# process restart cepki
Tue Dec  9 04:50:55.396 UTC
RP/0/RP0/CPU0:Dec  9 04:50:55.445 UTC: sysmgr_control[66932]:
%OS-SYSMGR-4-PROC_RESTART_NAME : User cafyauto (con0_RP0_CPU0) requested a restart of
process cepki at 0/RP0/CPU0
Router#


This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply third-party
authority to import, export, distribute or use encryption. Importers,
exporters, distributors and users are responsible for compliance with
U.S. and local country laws. By using this product you agree to comply
with applicable laws and regulations. If you are unable to comply with
U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be
found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html

If you require further assistance please contact us by sending email to
export@cisco.com.

RP/0/RP0/CPU0:Dec  9 04:50:56.430 UTC: cepki[277]:
%INFRA-WARN_INSECURE-4-INSECURE_FEATURE_WARN : Feature 'Weak RSA SSH host key present'
 utilized or configured. This feature is deprecated as it is known to be insecure; it
will be removed in a future release. Consider upgrading RSA (crypto key generate rsa)
to at least 3072 bits
```

- The router sends a syslog warning if the DSA keys are found during the system boot or process restart.

```
Router# process restart cepki
Tue Dec  9 05:14:49.926 UTC
RP/0/RP0/CPU0:Dec  9 05:14:49.973 UTC: sysmgr_control[68825]:
%OS-SYSMGR-4-PROC_RESTART_NAME : User cafyauto (con0_RP0_CPU0) requested a restart of
process cepki at 0/RP0/CPU0
Router#

This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply third-party
authority to import, export, distribute or use encryption. Importers,
exporters, distributors and users are responsible for compliance with
U.S. and local country laws. By using this product you agree to comply
with applicable laws and regulations. If you are unable to comply with
U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be
found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html

If you require further assistance please contact us by sending email to
export@cisco.com.

RP/0/RP0/CPU0:Dec  9 05:14:50.969 UTC: cepki[277]:
%INFRA-WARN_INSECURE-4-INSECURE_FEATURE_WARN : Feature 'Weak DSA SSH host key present'
 utilized or configured. This feature is deprecated as it is known to be insecure; it
will be removed in a future release. Consider removing DSA (crypto key zeroize dsa)
```

- The router sends a syslong if weak RSA keys are found during the system boot or process start.

```
Router# process restart cepki
Tue Dec  9 05:19:58.309 UTC
RP/0/RP0/CPU0:Dec  9 05:19:58.357 UTC: sysmgr_control[69248]:
%OS-SYSMGR-4-PROC_RESTART_NAME : User cafyauto (con0_RP0_CPU0) requested a restart of
process cepki at 0/RP0/CPU0
Router#

This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply third-party
authority to import, export, distribute or use encryption. Importers,
exporters, distributors and users are responsible for compliance with
U.S. and local country laws. By using this product you agree to comply
with applicable laws and regulations. If you are unable to comply with
U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be
found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html

If you require further assistance please contact us by sending email to
export@cisco.com.

RP/0/RP0/CPU0:Dec  9 05:19:59.319 UTC: cepki[277]:
%INFRA-WARN_INSECURE-4-INSECURE_FEATURE_WARN : Feature 'Weak RSA SSH host key present'
 utilized or configured. This feature is deprecated as it is known to be insecure; it
will be removed in a future release. Consider upgrading RSA (crypto key generate rsa)
to at least 3072 bits
```

Declaring a Certification Authority and Configuring a Trusted Point

This task declares a CA and configures a trusted point.



Warning If a `commit replace` operation removes a crypto CA trustpoint or certificate configuration, the system can clear the installed certificates when it processes the delete event. If the operation later fails and the configuration is restored, the certificates might not be restored. Services that use the certificates can become unavailable. After a failed `commit replace`, verify that the required certificates are still installed with **show crypto ca certificates** comamnd and reinstall any certificates that were cleared.

SUMMARY STEPS

1. **crypto ca trustpoint ca-name**
2. **enrollment url CA-URL**
3. **query url LDAP-URL**
4. **enrollment retry period minutes**
5. **enrollment retry count number**
6. **rsakeypair keypair-label**
7. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**crypto ca trustpoint ca-name** **Example:** `RP/0/RSP0/CPU0:router(config)# crypto ca trustpoint myca`	Declares a CA. • Configures a trusted point with a selected name so that your router can verify certificates issued to peers. • Enters trustpoint configuration mode.
Step 2	**enrollment url CA-URL** **Example:** `RP/0/RSP0/CPU0:router(config-trustp)# enrollment url http://ca.domain.com/certsrv/mscep/mscep.dll`	Specifies the URL of the CA. • The URL should include any nonstandard cgi-bin script location.
Step 3	**query url LDAP-URL** **Example:** `RP/0/RSP0/CPU0:router(config-trustp)# query url ldap://my-ldap.domain.com`	(Optional) Specifies the location of the LDAP server if your CA system supports the LDAP protocol.
Step 4	**enrollment retry period minutes** **Example:** `RP/0/RSP0/CPU0:router(config-trustp)# enrollment retry period 2`	(Optional) Specifies a retry period. • After requesting a certificate, the router waits to receive a certificate from the CA. If the router does not receive a certificate within a period of time (the retry period) the router will send another certificate request. • Range is from 1 to 60 minutes. Default is 1 minute.
Step 5	**enrollment retry count number** **Example:** `RP/0/RSP0/CPU0:router(config-trustp)# enrollment retry count 10`	(Optional) Specifies how many times the router continues to send unsuccessful certificate requests before giving up. • The range is from 1 to 100.
Step 6	**rsakeypair keypair-label** **Example:** `RP/0/RSP0/CPU0:router(config-trustp)# rsakeypair mykey`	(Optional) Specifies a named RSA key pair generated using the **crypto key generate rsa** command for this trustpoint. • Not setting this key pair means that the trustpoint uses the default RSA key in the current configuration.
Step 7	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session.

	Command or Action	Purpose
		• **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Authenticating the CA

This task authenticates the CA to your router.

The router must authenticate the CA by obtaining the self-signed certificate of the CA, which contains the public key of the CA. Because the certificate of the CA is self-signed (the CA signs its own certificate), manually authenticate the public key of the CA by contacting the CA administrator to compare the fingerprint of the CA certificate.

SUMMARY STEPS

1. **crypto ca authenticate ca-name**
2. show crypto ca certificates

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**crypto ca authenticate ca-name** **Example:** `RP/0/RSP0/CPU0:router# crypto ca authenticate myca`	Authenticates the CA to your router by obtaining a CA certificate, which contains the public key for the CA.
Step 2	show crypto ca certificates **Example:** `RP/0/RSP0/CPU0:router# show crypto ca certificates`	(Optional) Displays information about the CA certificate.

Multi-Tier Certificate Authority for Trustpoint Authentication

Table 21: Feature History Table

Feature Name	Release Information	Feature Description
Multi-Tier Certificate Authority for Trustpoint Authentication	Release 7.10.1	Apart from the root certificate authority (CA), you can now use a subordinate CA to issue certificates and authenticate your network devices. This feature is beneficial when you have an existing CA hierarchy where it is not the root CA but the subordinate CA that issues the leaf or router certificates. In earlier releases, you could associate only a single CA, not a multi-tier CA, to a trustpoint. And, you could use only the root CA certificate to enroll the router certificates.

During terminal-based enrollment of a CA trustpoint, Cisco IOS XR network devices accepted only Root CA certificate. You might have some network topologies which use multi-tier CA hierarchy for enrollment purposes because it provides more flexibility and security. From Cisco IOS XR Software Release 7.10.1 and later, as part of terminal-based authentication, you can import a complete CA hierarchy (from the Root CA till the subordinate CA that issues the certificate) as part of a single authentication request. With this feature, you can provide a certificate chain including the Root CA and intermediate subordinate CAs as part of the terminal-based enrollment process. This feature is useful when you have an existing CA hierarchy where it is not the root CA but the subordinate CA that issues the leaf or router certificates.

You can have a maximum of 8 tiers, that is, a chain of CA with one Root CA and seven subordinate CAs, for trustpoint authentication.

How to Use Multi-Tier CA for Trustpoint Authentication

Use the **crypto ca authenticate** command to use multi-tier CAs for trustpoint authentication or enrollment. You must use only privacy enhanced mail (PEM)-encoded certificates for trustpoint authentication using multi-tier CAs.

The enrollment process remains the same as that of the enrollment using single-tier CA, except that you get a message on the router console that prompts to use only PEM-encoded certificates.

Prerequisite

You must generate a key pair, import a public key, and configure a trustpoint on the router as detailed in the previous sections.

Configuration Example

```
Router#crypto ca authenticate test-ca
Mon Feb  6 08:17:48.943 UTC
```

```
Enter the base 64/PEM encoded certificate/certificates.
Please note: for multiple certificates use only PEM
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
MIIF5TCCA82gAwIBAgICEAEwDQYJKoZIhvcNAQELBQAwXTELMAkGA1UEBhMCSU4x
CzAJBgNVBAgMAktBMQwwCgYDVQQHDANCR0wxDTALBgNVBAoMBENTQ08xDTALBgNV
.
.
.
/4UzeeX6ll0gGJVbDwGeIZTH00artqxHquKQ2P7eXQ1pg0PRNRqWN90SvT5yE33N
eHgbtvdHg1K6K6IAj/NGnd7xUrA1TQ4bdmouCNkgbXM/G9DwgkOOvZ8KYRP9JW57
LYIv2ZcRS2vdnZRD9JPGVig2EgcfVPtj+Q==
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIF9TCCA92gAwIBAgIUD6AGesleqedhorkrJ9HWjz1RQzswDQYJKoZIhvcNAQEL
BQAwXTELMAkGA1UEBhMCSU4xCzAJBgNVBAgMAktBMQwwCgYDVQQHDANCR0wxDTAL
.
.
.
+6rMWd6BmfSy2PT3Qz5AjO2+3N1dd67qRRrX7skklkX4JXY42n5/l9PQtSp0wTBh
uy5yUAagynu0z07GczE7E9V+tJHRmNTbnd8pxLk4lTwqtiCIXwQLZA75SkwCS5wh
fn7OrV7uFjMaggNkvj0kSSOkWxqJ+j/KqMAA2zQMUV+qdvT6i+ZV44U=
-----END CERTIFICATE-----
Serial Number  : 10:01
  Subject:
 CN=SUB_CA_CERT,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Issued By      :
 CN=TWO-LEVEL-CA,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Validity Start : 12:31:40 UTC Sun Jun 14 2020
  Validity End   : 12:31:40 UTC Wed Jun 12 2030

  CRL Distribution Point
 http://10.105.236.78/crl_akshath_two_level_ca/crl.der
  SHA1 Fingerprint:
  D8E0C11ECED96F67FDBC800DB6A126676A76BD62
  Serial Number  : 0F:A0:06:7A:C9:5E:A9:E7:61:A2:B9:2B:27:D1:D6:8F:3D:51:43:3B
  Subject:
 CN=TWO-LEVEL-CA,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Issued By      :
 CN=TWO-LEVEL-CA,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Validity Start : 13:12:32 UTC Sun Jun 07 2020
  Validity End   : 13:12:32 UTC Sat Jun 02 2040

  CRL Distribution Point
 http://10.105.236.78/crl_akshath_two_level_ca/crl.der
  SHA1 Fingerprint:
  08E71248FB7578614442E713AC87C461D173952F

CA Certificate validated using issuer certificate.
Router#
```

Verification

Use the **show crypto ca certificates** *trustpoint-name* to view the CA certificate chain. The command output displays the **Trusted Certificate Chain** field if there is one or more subordinate CAs involved in the hierarchy.

```
Router#show crypto ca certificates test-ca
Mon Feb  6 09:03:53.019 UTC

Trustpoint       : test-ca
==================================================
```

```
CA certificate
  Serial Number  : 10:01
  Subject:
            CN=SUB_CA_CERT,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Issued By      :
            CN=TWO-LEVEL-CA,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Validity Start : 12:31:40 UTC Sun Jun 14 2020
  Validity End   : 12:31:40 UTC Wed Jun 12 2030

  CRL Distribution Point
            http://10.105.236.78/crl_akshath_two_level_ca/crl.der
  SHA1 Fingerprint:
             D8E0C11ECED96F67FDBC800DB6A126676A76BD62
Trusted Certificate Chain
  Serial Number  : 0F:A0:06:7A:C9:5E:A9:E7:61:A2:B9:2B:27:D1:D6:8F:3D:51:43:3B
  Subject:
            CN=TWO-LEVEL-CA,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Issued By      :
            CN=TWO-LEVEL-CA,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Validity Start : 13:12:32 UTC Sun Jun 07 2020
  Validity End   : 13:12:32 UTC Sat Jun 02 2040

  CRL Distribution Point
            http://10.105.236.78/crl_akshath_two_level_ca/crl.der
  SHA1 Fingerprint:
             08E71248FB7578614442E713AC87C461D173952F
Router certificate
  Key usage      : General Purpose
  Status         : Available
  Serial Number  : 28:E5
  Subject:
            CN=test
  Issued By      :
            CN=SUB_CA_CERT,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Validity Start : 08:49:54 UTC Mon Feb 06 2023
  Validity End   : 08:49:54 UTC Wed Mar 08 2023
  SHA1 Fingerprint:
             6C8644FA67D9CEBC7C5665C35838265F578835AB
Associated Trustpoint: test-ca
```

Requesting Your Own Certificates

This task requests certificates from the CA.

You must obtain a signed certificate from the CA for each of your router's RSA key pairs. If you generated general-purpose RSA keys, your router has only one RSA key pair and needs only one certificate. If you previously generated special usage RSA keys, your router has two RSA key pairs and needs two certificates.

SUMMARY STEPS

1. **crypto ca enroll ca-name**
2. **show crypto ca certificates**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**crypto ca enroll ca-name** **Example:** `RP/0/RSP0/CPU0:router# crypto ca enroll myca`	Requests certificates for all of your RSA key pairs. • This command causes your router to request as many certificates as there are RSA key pairs, so you need only perform this command once, even if you have special usage RSA key pairs. • This command requires you to create a challenge password that is not saved with the configuration. This password is required if your certificate needs to be revoked, so you must remember this password. • A certificate may be issued immediately or the router sends a certificate request every minute until the enrollment retry period is reached and a timeout occurs. If a timeout occurs, contact your system administrator to get your request approved, and then enter this command again.
Step 2	**show crypto ca certificates** **Example:** `RP/0/RSP0/CPU0:router# show crypto ca certificates`	(Optional) Displays information about the CA certificate.

Configuring Certificate Enrollment Using Cut-and-Paste

This task declares the trustpoint certification authority (CA) that your router should use and configures that trustpoint CA for manual enrollment by using cut-and-paste.

SUMMARY STEPS

1. **configure**
2. **crypto ca trustpoint** *ca-name*
3. enrollment terminal
4. Use the **commit** or **end** command.
5. **crypto ca authenticate** *ca-name*
6. **crypto ca enroll** *ca-name*
7. **crypto ca import** ca- *name* **certificate**
8. show crypto ca certificates

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**crypto ca trustpoint** *ca-name*
	Example:
	`RP/0/RSP0/CPU0:router(config)# crypto ca trustpoint myca`
	`RP/0/RSP0/CPU0:router(config-trustp)#`
		Declares the CA that your router should use and enters trustpoint configuration mode.
		• Use the *ca-name* argument to specify the name of the CA.

Step 3	enrollment terminal **Example:** `RP/0/RSP0/CPU0:router(config-trustp)# enrollment terminal`	Specifies manual cut-and-paste certificate enrollment.
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 5	**crypto ca authenticate** *ca-name* **Example:** `RP/0/RSP0/CPU0:router# crypto ca authenticate myca`	Authenticates the CA by obtaining the certificate of the CA. • Use the *ca-name* argument to specify the name of the CA. Use the same name that you entered in Step 2, on page 151.
Step 6	**crypto ca enroll** *ca-name* **Example:** `RP/0/RSP0/CPU0:router# crypto ca enroll myca`	Obtains the certificates for your router from the CA. • Use the *ca-name* argument to specify the name of the CA. Use the same name that you entered in Step 2.
Step 7	**crypto ca import** ca- *name* **certificate** **Example:** `RP/0/RSP0/CPU0:router# crypto ca import myca certificate`	Imports a certificate manually at the terminal. • Use the *ca-name* argument to specify the name of the CA. Use the same name that you entered in Step 2.

	Command or Action	Purpose
		Note You must enter the **crypto ca import** comm used. The first time the command is entered, the command is entered, the other certificate pasted first.)
Step 8	show crypto ca certificates **Example:** `RP/0/RSP0/CPU0:router# show crypto ca certificates`	Displays information about your certificate and the CA certificate.

Configuration Examples for Implementing Certification Authority Interoperability

This section provides the following configuration example:

Configuring Certification Authority Interoperability: Example

The following example shows how to configure CA interoperability.

Comments are included within the configuration to explain various commands.

```
configure
hostname myrouter
domain name mydomain.com
end

Uncommitted changes found, commit them? [yes]:yes

crypto key generate rsa mykey

The name for the keys will be:mykey
Choose the size of the key modulus in the range of 360 to 2048 for your General Purpose
Keypair
Choosing a key modulus greater than 512 may take a few minutes.
How many bits in the modulus [1024]:
Generating RSA keys ...
Done w/ crypto generate keypair
[OK]

show crypto key mypubkey rsa

Key label:mykey
Type     :RSA General purpose
Size     :1024
Created  :17:33:23 UTC Thu Sep 18 2003
Data     :
 30819F30 0D06092A 864886F7 0D010101 05000381 8D003081 89028181 00CB8D86
 BF6707AA FD7E4F08 A1F70080 B9E6016B 8128004C B477817B BCF35106 BC60B06E
```

```
 07A417FD 7979D262 B35465A6 1D3B70D1 36ACAFBD 7F91D5A0 CFB0EE91 B9D52C69
 7CAF89ED F66A6A58 89EEF776 A03916CB 3663FB17 B7DBEBF8 1C54AF7F 293F3004
 C15B08A8 C6965F1E 289DD724 BD40AF59 E90E44D5 7D590000 5C4BEA9D B5020301
 0001

! The following commands declare a CA and configure a trusted point.

configure
crypto ca trustpoint myca
enrollment url http://xyz-ultra5
enrollment retry count 25
enrollment retry period 2
rsakeypair mykey
end

Uncommitted changes found, commit them? [yes]:yes

! The following command authenticates the CA to your router.

crypto ca authenticate myca

Serial Number  :01
Subject Name   :
cn=Root coax-u10 Certificate Manager,ou=HFR,o=Cisco Systems,l=San Jose,st=CA,c=US
Issued By      :
cn=Root coax-u10 Certificate Manager,ou=HFR,o=Cisco Systems,l=San Jose,st=CA,c=US
Validity Start :07:00:00 UTC Tue Aug 19 2003
Validity End   :07:00:00 UTC Wed Aug 19 2020
Fingerprint:58 71 FB 94 55 65 D4 64 38 91 2B 00 61 E9 F8 05
Do you accept this certificate?? [yes/no]:yes

! The following command requests certificates for all of your RSA key pairs.

crypto ca enroll myca

% Start certificate enrollment ...
% Create a challenge password. You will need to verbally provide this
  password to the CA Administrator in order to revoke your certificate.
% For security reasons your password will not be saved in the configuration.
% Please make a note of it.

Password:
Re-enter Password:
    Fingerprint: 17D8B38D ED2BDF2E DF8ADBF7 A7DBE35A

! The following command displays information about your certificate and the CA certificate.

show crypto ca certificates

Trustpoint       :myca
============================================================
CA certificate
  Serial Number  :01
  Subject Name   :
        cn=Root coax-u10 Certificate Manager,ou=HFR,o=Cisco Systems,l=San Jose,st=CA,c=US
  Issued By      :
        cn=Root coax-u10 Certificate Manager,ou=HFR,o=Cisco Systems,l=San Jose,st=CA,c=US
  Validity Start :07:00:00 UTC Tue Aug 19 2003
  Validity End   :07:00:00 UTC Wed Aug 19 2020
Router certificate
  Key usage      :General Purpose
  Status         :Available
  Serial Number  :6E
  Subject Name   :
```

```
      unstructuredName=myrouter.mydomain.com,o=Cisco Systems
Issued By      :
      cn=Root coax-u10 Certificate Manager,ou=HFR,o=Cisco Systems,l=San Jose,st=CA,c=US
Validity Start :21:43:14 UTC Mon Sep 22 2003
Validity End   :21:43:14 UTC Mon Sep 29 2003
CRL Distribution Point
      ldap://coax-u10.cisco.com/CN=Root coax-u10 Certificate Manager,O=Cisco Systems
```

Expiry Notification for PKI Certificate

The section provides information about the notification mechanism using SNMP trap and syslog messages when a public key infrastructure (PKI) certificate is approaching its expiry date.

Learn About the PKI Alert Notification

Security is critical and availability of certificates for applications is vital for authenticating the router. If the certificate expires, they become invalid and impacts services like Crosswork Trust Insights, Internet Key Exchange version 2, dot1x, and so on.

What if there is a mechanism to alert the user about the expiry date of the certificate?

From Release 7.1.1, IOS -XR provides a mechanism by which a CA client sends a notification to a syslog server when certificates are on the verge of expiry. Alert notifications are sent either through the syslog server or Simple Network Management Protocol (SNMP) traps.

PKI traps retrieves the certificate information of the devices in the network. The device sends SNMP traps at regular intervals to the network management system (NMS) based on the threshold configured in the device.

An SNMP trap (certificate expiry notification) is sent to the SNMP server at regular intervals starting from 60 days to one week before the certificate end date. The notifications are sent at the following intervals:

The notifications are sent at the following intervals:

Intervals	Description	Notification Mode
First notification	The notification is sent 60 days before the expiry of the certificate.	The notificiation are in a warning mode.
Repeated notifications	The repeated notification is sent every week, until a week before the expiry of the certificate. The notifications are in a warning mode when the certificate is valid for more than a week.	The notifications are in a warning mode when the certificate is valid for more than a week.
Last notification	The notifications are sent every day until the certificate expiry date.	The notifications are in an alert mode when the validity of a certificate is less than a week.

The notifications include the following information:

- Certificate serial number
- Certificate issuer name

- Trustpoint name
- Certificate type
- Number of days remaining for the certificate to expire
- Certificate subject name

The following is a syslog message that is displayed on the device:

```
%SECURITY-CEPKI-1-CERT_EXPIRING_ALERT : Certificate expiring WITHIN A WEEK.
Trustpoint Name= check, Certificate Type= ID, Serial Number= 02:EC,
Issuer Name= CN=cacert,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN, Subject name= CN=cisco.com,
Time Left= 1 days, 23 hours, 59 minutes, 41 seconds
```

Restrictions for PKI Credentials Expiry Alerts

Alerts are not sent for the following certificates:

- Secure Unique Device Identifier (SUDI) certificates
- Certificates that belong to a trustpool. Trustpools have their own expiry alerts mechanism
- Trustpoint clones
- CA certificates that do not have a router certificate associated with it.
- Certificates with key usage keys

Enable PKI Traps

This feature cannot be disabled and requires no additional configuration tasks.

To enable PKI traps, use the **snmp-server traps pki** command. If SNMP is configured, the SNMP trap is configured in the same PKI expiry timer.

```
Router(config)# snmp-server traps pki
Router(config)# commit
```

Verification

This example shows sample output from the show running-config command.

```
Router# show runn snmp-server traps
snmp-server traps pki
```

What's Next: See Regenerate the Certificate, on page 155.

Regenerate the Certificate

The certificate becomes invalid once expired. When you see the certificate expiry notification, we recommend you to regenerate the certificate, as soon as possible.

Perform the following steps, to regenerate the certificates:

1. Clear the existing certificate using the following command:

   ```
   Router# clear crypto ca certificates [trustpoint-name]
   ```

For example,

```
Router# clear crypto ca certificates myca
```

2. We recommend you to regenerate a new keypair for the label configured under the trustpoint-name. The new keypair overwrites the old key pair.

```
Router# crypto key generate rsa [keypair-lablel]
```

For example,

```
Router# crypto key generate rsa mykey
The name for the keys will be: mykey
% You already have keys defined for mykey
Do you really want to replace them? [yes/no]: yes
  Choose the size of the key modulus in the range of 512 to 4096 for your General Purpose
 Keypair. Choosing a key modulus greater than 512 may take a few minutes.

How many bits in the modulus [2048]:
Generating RSA keys ...
Done w/ crypto generate keypair
[OK]The name for the keys will be: mykey
% You already have keys defined for mykey
Do you really want to replace them? [yes/no]: yes
  Choose the size of the key modulus in the range of 512 to 4096 for your General Purpose
 Keypair. Choosing a key modulus greater than 512 may take a few minutes.

How many bits in the modulus [2048]:
Generating RSA keys ...
Done w/ crypto generate keypair
[OK]
```

3. Reenroll the certificate using the following command. For more information, see Requesting Your Own Certificates, on page 149.

```
Router# crypto ca authenticate [trustpoint-name]
Router# crypto ca enroll [trustpoint-name]
```

For example,

```
Router# crypto ca authenticate myca
Router# crypto ca enroll myca
```

Automatic renewal of PKI certificate

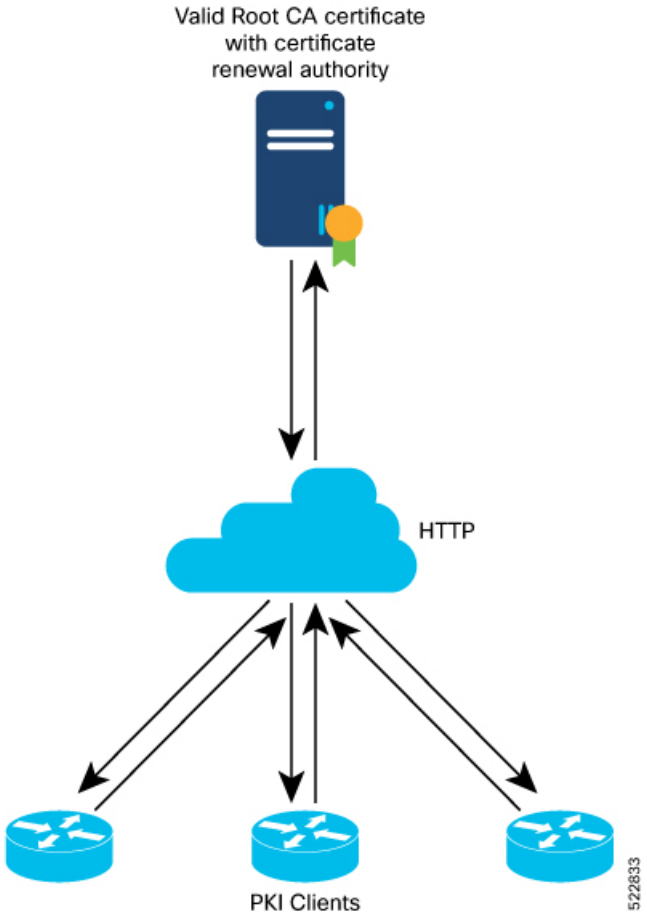
Table 22: Feature History Table

Feature Name	Release Information	Feature Description
Automatic renewal of Public Key Infrastructure (PKI) certificate	Release 7.5.3	You can now enable the router to renew the PKI certificate from the Certificate Authority (CA) by configuring the percentage of the certificate validity, after which the router requests a new certificate from the CA, and the CA authorizes it before certification expiration. This feature eliminates the previously needed manual efforts of certification renewal and avoids interruptions such as MACsec session flaps due to certificate expiry and so on. This feature introduces the following commands: • **auto-enroll** • **renewal-message-type**

The public key infrastructure (PKI) controls the digital certificates that protect the sensitive information flowing through a network to provide secure end-to-end communication. The PKI encrypts and decrypts data using a public key and a private key pair that it generates. A PKI (digital) certificate is a digital entity used to authenticate the identity of a router. These PKI certificates often have a short validity time and would need manual efforts from the network operators to replace them in time.

With automatic renewal of PKI certificate, the router has an ability to automatically renew the PKI certificate when it is approaching its expiry date. Here, you can configure a timeline for the PKI certificate renewal. You can specify the percentage of certificate validity, after which you would prefer the router to request a new certificate from the CA server. This timeline for auto-renewal of the PKI certificate is called auto-enroll. When a router with auto-enroll configured completes the said period, the router will generate a certificate signing request and sends the request to the CA using Simple Certificate Management Protocol (SCMP). The CA server will immediately create a newly signed certificate and ensures that it replaces the old certificate in the router. This way, automatic renewal of the PKI certificate before expiry avoids any interruptions to data flowing through the network.

Figure 5: Automatic renewal of Public Key Infrastructure (PKI) certificate



Pre-requisites

- Ensure that the CA has a valid certificate.
- Make sure that the CA possess certificate renewal capability.
- You must configure a trustpoint in the router. Configure the trustpoint using **crypto ca trustpoint** command.

Note A trustpoint should be authenticated before any enrolment. The trustpoint is authenticated when it has a CA certificate, and it is enrolled when it has a router certificate.

- The communication between the PKI Client and CA server should be over HTTP protocol. That is, the enrollment url for CA server should a HTTP url.

Configuration Guidelines

- The PKI certificates are signed using the RSA algorithm only.
- If you configure the auto-enroll option under trustpoint after the renew timer for a PKI certificate has started, then such configuration will only apply to the next renewal cycle and not the current one. The same condition applies while configuring the **no auto-enroll** option as well.
- The auto-enroll percentage may range between 1 and 99.
- The certificate renewal process requires the serial number and IP address values in the trustpoint. If these values are readily available under the trustpoint, the renewal process obtains it from there. If not, the router CLI requests you to configure these values during trustpoint enrollment.
- By default, the PKI uses PKCS requests for automatic certification renewal. You can also configure the router to use the Renewal request by executing the **renewal-message-type renewalreq** command.
- If the CA server is unable to address the certificate renewal requests, it requests the router to poll the renewal request. In such scenarios, by default, the router retries for 10 minutes with a gap of 1 minute between each request if a certificate renewal attempt fails. You can also configure these values using the **enrollment retry count** and **enrollment retry period** commands.

Configuration Example

Configuration

Configuring this feature using the CLI:

```
Router# configure
Router(config)# crypto ca trustpoint test
Router(config-trustp)# enrollment url http://frog.phoobin.com
Router(config-trustp)# subject-name OU=Spiral Dept., O=tiedye.com
Router(config-trustp)# auto-enroll 30
Router(config-trustp)# commit
```

To disable this feature, execute the following:

```
Router# configure
Router(config)# no auto-enroll
Router(config-trustp)# commit
```

Running configuration

```
Router# show running-config crypto ca trustpoint test
crypto ca trustpoint test
 enrollment url http://frog.phoobin.com
 auto-enroll 30
!
```

Integrating Cisco IOS XR and Crosswork Trust Insights

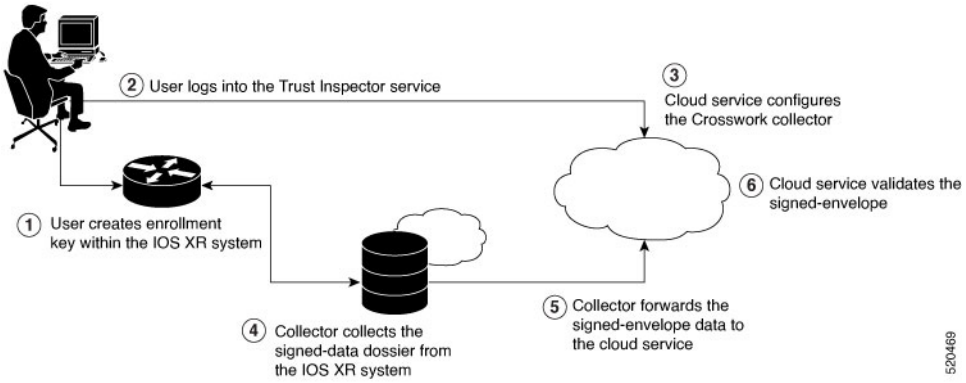
The Cisco IOS XR 64-bit Software provides you the infrastructure to enroll and share the signed-data with Cisco Crosswork cloud infrastructure and applications. The Cisco Crosswork Trust Insights is a cloud-based Software as a service (SaaS) that provides signed and encrypted system integrity information to track the trust

posture of network hardware and software components. For details, see Cisco Crosswork Trust Insights Data Sheet.

Integrating IOS XR and Crosswork Trust Insights include these main processes:

- System enrollment – Enrolling a Cisco IOS XR platform into Crosswork cloud infrastructure.
- Signed-data sharing – Sharing the data for infrastructure trust analysis between the systems that run IOS XR and Crosswork. This involves collecting the signed-data dossier, that is, signed-data that is needed for infrastructure trust inspection service.

Workflow



The following steps depict the workflow of Cisco IOS XR and Crosswork Trust Insights integration:

1. As part of the enrollment process, the user generates new key pair and trust root within the IOS XR system by using the IOS XR commands.
2. The user logs into the Trust Inspector service, and enters the enrollment workflow in the enrollment dialog to create a new device ID. The user must provide the management IP address, login credentials and certificate root to the Trust Inspector service.
3. The Trust Inspector service configures the Crosswork collector to log in to the router, and to pull the data that is pushed down from the cloud to the collector

   .
4. The Crosswork collector begins a periodic polling cycle and executes a command to generate a signed-information dossier from each IOS XR instance that is being polled.
5. The collector forwards the signed-envelope data to the cloud service for validation.
6. The cloud service validates signed-envelope against the enrolled certificate or trust chain.

How to Integrate Cisco IOS XR and Crosswork Trust Insights

Integrating Cisco IOS XR and Crosswork Trust Insights involve these main tasks for system enrollment and signed-data sharing:

- Generate Key Pair, on page 163
- Generate System Trust Point for the Leaf and Root Certificate, on page 164

- Generate Root and Leaf Certificates, on page 165
- System Certificates Expiry, on page 167
- Collect Data Dossier, on page 167

For details of IOS XR commands used in this configuration, see the *Public Key Infrastructure Commands* chapter in the *System Security Command Reference for Cisco ASR 9000 Series Routers*.

Prerequisites

Before you begin, you must check here for any available IOS XR Software Maintenance Updates (SMUs) specific to Crosswork Trust Insights. For information related to SMUs, see Cisco IOS XR Release Notes.

You must ensure that the below configurations are present on the IOS XR device, before starting IOS XR and Crossworks Trust Insights integration.

- User authorization required to collect the signed-data dossier
- SSH server configuration
- Netconf server configuration
- Domain name configuration, which is required for certification enrollment

The sections given below lists the configuration example for the prerequisites.

Configuration Example for User Authorization

You must have the required user access privileges in order to collect the data dossier from the system. This is defined in terms of IOS XR Task IDs for each command.

For the respective Task ID applicable for each data dossier option and for the signed-envelope, see the Task ID section in the Command Reference page of **show platform security integrity dossier** command and **utility sign** command.

Listed below are the configurations to set up a user with sufficient authorization to collect all the signed-data dossier. You can configure customized task groups, then associate those task groups with user groups, and finally associate the user groups with the user.



Note We recommend that you use the **task execute dossier** to configure a CTI (customer-define) user, who collects dossier from the system.

```
Router#configure
Router(config)#taskgroup alltasks-dossier
Router(config-tg)#task read sysmgr
Router(config-tg)#task read system
Router(config-tg)#task read pkg-mgmt
Router(config-tg)#task read basic-services
Router(config-tg)#task read config-services
Router(config-tg)#task execute dossier
Router(config-tg)#task execute basic-services
Router(config-tg)#commit
```

```
Router#configure
Router(config)#usergroup dossier-group
Router(config-ug)#taskgroup alltasks-dossier
Router(config-ug)#commit
```

```
Router#configure
Router(config)#username dossier-user
Router(config-un)#group dossier-group
Router(config-un)#commit
```

Configuration Example for SSH and Netconf

```
Router#configure
Router(config)#ssh server v2
Router(config)#ssh server vrf default
Router(config)#ssh server netconf vrf default
Router(config)#netconf-yang agent
Router(config-ncy-agent)#ssh
Router(config-ncy-agent)#exit
Router(config)#domain name example.com
Router(config)#commit
```

Running Configuration for SSH and Netconf

```
ssh server v2
ssh server vrf default
ssh server netconf vrf default
!
netconf-yang agent
 ssh
!
domain name example.com
```

While the dossier is collected from a device through SSH, the SSH session might timeout. Also, multiple ssh sessions to a device can result in the denial of some SSH sessions. To avoid such occurrence, the following configuration is recommended on the device:

```
Router#configure
Router(config)#ssh server rate-limit 600
Router(config)#line default
Router(config-line)#exec-timeout 0 0
Router(config-line)#session-timeout 0
Router(config-line)#commit
```

Running Configuration

```
ssh server rate-limit 600
!
line default
 exec-timeout 0 0
 session-timeout 0
!
```

Generate Key Pair

To enroll a system running Cisco IOS XR Software, you must generate the key and the certificate for both the leaf and the root node. The system supports a two tier self-signed certificate chain for the enrollment key to support re-keying without re-enrollment of the certificate with the Crossworks service.

You can use the **system-root-key** and **system-enroll-key** options in the **crypto key generate** command to generate the root key and the enrollment key respectively, for all the hashing algorithms. You can do this for hashing algorithms such as RSA, DSA or ECDSA (including ECDSA nistp384 and ECDSA nitsp521).

Example of Generating Key Pair

Key pair generation for root:

```
Router#crypto key generate rsa system-root-key

Sun Oct 20 13:05:26.657 UTC
The name for the keys will be: system-root-key
Choose the size of the key modulus in the range of 512 to 4096 for your General Purpose
Keypair. Choosing a key modulus greater than 512 may take a few minutes.
How many bits in the modulus [2048]:
Generating RSA keys ...
Done w/ crypto generate keypair
[OK]
```

Key pair generation for leaf:

```
Router#crypto key generate rsa system-enroll-key

Sun Oct 20 13:05:40.370 UTC
The name for the keys will be: system-enroll-key
Choose the size of the key modulus in the range of 512 to 4096 for your General Purpose
Keypair. Choosing a key modulus greater than 512 may take a few minutes.
 How many bits in the modulus [2048]:
Generating RSA keys ...
Done w/ crypto generate keypair
[OK]
```

Verification

You can use the **show crypto key mypubkey rsa** command to verify the above key pair generation.

```
Router#show  crypto  key mypubkey rsa | begin system-
Fri Mar 27 14:00:20.954 IST
Key label: system-root-key
Type     : RSA General purpose
Size     : 2048
Created  : 01:13:10 IST Thu Feb 06 2020
Data     :
 30820122 300D0609 2A864886 F70D0101 01050003 82010F00 3082010A 02820101
 00A93DE0 1E485EE3 0E7F0964 C48361D1 B6014BE7 A303D8D6 F7790E92 88E69C4B
 B97B7A9C D1B277E3 1569093C 82BD3258 7F67FB49 94860ECD 34498F1F 59B45757
 F32C8E8F 7CEE23EC C36A43D1 9F85C0D9 B96A14DD DD3BBD4C A1FB0888 EED210A7
 39D9A403 7ACE0F6E 39107226 CA621AD8 6E8102CA 9761B86F D33F2871 9DD16559
 AFCB4729 EFCEDBAF 83DF76E4 9A439844 EE3B1180 4022F575 99E11A2C E25BB23D
 9DD74C81 4E5C1345 D9E3CC79 1B98B1AA 6C06F004 22B901EC 36C099FE 10DE2622
 EB7CE618 9A555769 12D94C90 D9BEE5EA A664E7F6 4DF8D8D4 FE7EAB07 1EF4FEAB
```

```
 22D9E55F 62BA66A0 72153CEC 81F2639F B5F2B5C5 25E10364 19387C6B E8DB8990
 11020301 0001

Key label: system-enroll-key
Type     : RSA General purpose
Size     : 2048
Created  : 01:13:16 IST Thu Feb 06 2020
Data     :
 30820122 300D0609 2A864886 F70D0101 01050003 82010F00 3082010A 02820101
 009DBC14 C83604E4 EB3D3CF8 5BA7FDDB 80F7E85B 427332D8 BBF80148 F0A9C281
 49F87D5C 0CEBA532 EBE797C5 7F174C69 0735D13A 493670CB 63B04A12 4BCA7134
 EE0031E9 047CAA1E 802030C5 6071E8C2 F8ECE002 CC3B54E7 5FD24E5C 61B7B7B0
 68FA2EFA 0B83799F 77AE4621 435D9DFF 1D713108 37B614D3 255020F9 09CD32E8
 82B07CD7 01A53896 6DD92B5D 5119597C 98D394E9 DBD1ABAF 6DE949FE 4A8BF1E7
 851EB3F4 60B1114A 1456723E 063E50C4 2D410906 BDB7590B F1D58480 F3FA911A
 6C9CD02A 58E68D04 E94C098F 0F0E81DB 76B40C55 64603499 2AC0547A D652412A
 BCBBF69F 76B351EE 9B2DF79D E490C0F6 92D1BB97 B905F33B FAB53C20 DDE2BB22
 C7020301 0001
```

Associated Commands

- **crypto key generate dsa**
- **crypto key generate ecdsa**
- **crypto key generate rsa**
- **show crypto key mypubkey dsa**
- **show crypto key mypubkey ecdsa**
- **show crypto key mypubkey rsa**

Generate System Trust Point for the Leaf and Root Certificate

You must configure these steps to generate the system trust point for the root and the leaf certificate:

Configuration Example

```
Router#config
Router(config)#domain name domain1
Router(config)#crypto ca trustpoint system-trustpoint
Router(config)#keypair rsa system-enroll-key
Router(config)#ca-keypair rsa system-root-key
Router(config)#subject-name CN=lab1-ads,C=US,ST=CA,L=San Jose,O=cisco systems,OU=ASR
Router(config)#subject-name ca-certificate CN=lab1-ca,C=US,ST=CA,L=San Jose,O=cisco
systems,OU=ASR
Router(config)#enrollment url self
Router(config)#key-usage certificate digitalsignature keyagreement dataencipherment
Router(config)#lifetime certificate 300
Router(config)#message-digest sha256
Router(config)#key-usage ca-certificate digitalsignature keycertsign crlsign
Router(config)#lifetime ca-certificate 367
Router(config)#commit
```

Running Configuration

```
config
domain name domain1
crypto ca trustpoint system-trustpoint
keypair rsa system-enroll-key
ca-keypair rsa system-root-key
subject-name CN=lab1-ads,C=US,ST=CA,L=San Jose,O=cisco systems,OU=ASR
subject-name ca-certificate CN=lab1-ca,C=US,ST=CA,L=San Jose,O=cisco systems,OU=ASR
enrollment url self
key-usage certificate digitalsignature keyagreement dataencipherment
lifetime certificate 300
message-digest sha256
key-usage ca-certificate digitalsignature keycertsign crlsign
lifetime ca-certificate 367
!
```

Associated Commands

- ca-keypair
- crypto ca trustpoint
- domain
- enrollment
- key-usage
- key-pair
- lifetime
- message-digest
- subject-name

Generate Root and Leaf Certificates

You must perform these steps to generate the root and the leaf certificates.

The root certificate is self-signed. The root certificate signs the leaf certificate.

Example of Generating Root Certificate

```
Router#crypto ca authenticate system-trustpoint

Sun Oct 20 13:07:24.136 UTC
% The subject name in the certificate will include: CN=lab1
ca,C=US,ST=CA,L=San Jose,O=cisco systems,OU=ASR
% The subject name in the certificate will include: ios.cisco.com
Serial Number  : 0B:62
Subject:
serialNumber=c44a11fc,unstructuredName=ios.cisco.com,OU=ASR,O=cisco systems,L=San
Jose,ST=CA,C=US,CN=lab1-ca
Issued By      :
        serialNumber=c44a11fc,unstructuredName=ios.cisco.com,OU=ASR,O=cisco systems,L=San
Jose,ST=CA,C=US,CN=lab1-ca
  Validity Start : 13:07:26 UTC Sun Oct 20 2019
  Validity End   : 13:07:26 UTC Wed Oct 21 2020
  SHA1 Fingerprint:
         9DD50A6B24FEBC1DDEE40CD2B4D99A829F260967
```

Example of Generating Leaf Certificate

```
Router#crypto ca enroll system-trustpoint

Sun Oct 20 13:07:45.593 UTC
% The subject name in the certificate will include: CN=lab1-ads,C=US,ST=CA,L=San Jose,O=cisco
 systems,OU=ASR
% The subject name in the certificate will include: ios.cisco.com
% Include the router serial number in the subject name? [yes/no]: yes
% The serial number in the certificate will be: c44a11fc
% Include an IP address in the subject name? [yes/no]: no
Certificate keypair configured Type: 1, Label: system-enroll-key.Leaf cert key usage string:
 critical,digitalSignature,keyEncipherment,keyAgreement.  Serial Number  : 0B:63
  Subject:
        serialNumber=c44a11fc,unstructuredName=ios.cisco.com,OU=ASR,O=cisco systems,L=San
Jose,ST=CA,C=US,CN=lab1-ads
  Issued By      :
        serialNumber=c44a11fc,unstructuredName=ios.cisco.com,OU=ASR,O=cisco systems,L=San
Jose,ST=CA,C=US,CN=lab1-ca
  Validity Start : 13:07:47 UTC Sun Oct 20 2019
  Validity End   : 13:07:47 UTC Sat Aug 15 2020
  SHA1 Fingerprint:
         19D4C40F9EFF8FF25B59DE0161BA6C0706DC9E3A
```

Verification

You can use the **show crypto ca certificates system-trustpoint [detail]** command to see the details of generated root and leaf certificates:

```
Router#show crypto ca certificates system-trustpoint
Fri Mar 27 14:00:51.037 IST

Trustpoint       : system-trustpoint
==================================================
CA certificate
  Serial Number  : 10:B5
  Subject:
        serialNumber=7b20faa4,unstructuredName=test-sec1.cisco.com
  Issued By      :
        serialNumber=7b20faa4,unstructuredName=test-sec1.cisco.com
  Validity Start : 12:30:17 UTC Fri Feb 21 2020
  Validity End   : 12:30:17 UTC Sat Feb 20 2021
  SHA1 Fingerprint:
         9400A30816805219FAAA5B9C86C214E6F34CEF7B
Router certificate
  Key usage      : General Purpose
  Status         : Available
  Serial Number  : 10:B6
  Subject:

serialNumber=7b20faa4,unstructuredAddress=10.1.1.1,unstructuredName=test-sec1.cisco.com,CN=Anetwork,OU=IT,O=Spark
 Network,L=Rotterdam,ST=Zuid Holland,C=NL
  Issued By      :
        serialNumber=7b20faa4,unstructuredName=test-sec1.cisco.com
  Validity Start : 12:30:31 UTC Fri Feb 21 2020
  Validity End   : 12:30:31 UTC Sat Feb 20 2021
  SHA1 Fingerprint:
         21ACDD5EB6E6F4103E02C1BAB107AD86DDCDD1F3
Associated Trustpoint: system-trustpoint
```

Associated Commands

- **crypto ca authenticate**
- **crypto ca enroll**
- **show crypto ca certificates system-trustpoint**

System Certificates Expiry

You need to regenerate the certificate, before it expires. From Release 7.1.1, IOS -XR provides a mechanism by which a CA client sends a notification to a syslog server when certificates are on the verge of expiry. For more information see Learn About the PKI Alert Notification, on page 154.

When you see the certificate expiry notification, we recommend you to regenerate the certificate, see Regenerate the Certificate, on page 155.

The following example shows how to regenerate the certificate.

```
Router# clear crypto ca certificates system-trustpoint
Router# crypto ca authenticate system-trustpoint
Router# crypto ca enroll system-trustpoint
```

Collect Data Dossier

Table 23: Feature History Table

Feature Name	Release Information	Description
Collect Filesystem Inventory	Release 7.3.1	With this feature, a snapshot of the filesystem metadata such as when the file was created, modified, or accessed is collected at each configured interval. In addition to displaying the changes that the file underwent as compared to the previous snapshot, the inventory helps in maintaining data integrity of all the files in the system.

Feature Name	Release Information	Description
IMA Optimization	Release 7.3.1	Integrity Measurement Architecture (IMA) is a Linux-based utility that attests and appraises the integrity of a system security, at runtime. In this release, IMA introduces the following IMA optimization aspects: • Incremental IMA that collects IMA events selectively and progressively instead of collecting all the IMA events at the same time. You can define the start of an IMA sequence, which consists of start event, start sequence number, and start time. • SUDI Signature - provides the hardware root of trust to the dossier that is collected by the system.
Support for Display Compact Option	Release 7.4.1	This release introduces: • The display compact option in the dossier CLI. The dossier contains all the fields of the IMA events, thus making the file size very heavy. With the display compact option, the system allows you to obtain IMA event logs in the protobuf format, which can then be decoded at the client end. The `display compact` option is added to the `show platform security integrity dossier include system-integrity-snapshot` command.

The Cisco IOS XR Software provides a data dossier command, **show platform security integrity dossier**, that helps in collecting the data from various IOS XR components. The output is presented in JSON format.

You can choose various selectors for this command as given below:

```
Router#show platform security integrity dossier include packages reboot-history
```

```
rollback-history system-integrity-snapshot filesystem-inventory system-inventory nonce 1580
 | utility sign nonce 1580 include-certificate
```

Create Signed-Envelope

To verify the data integrity and authenticity of the data dossier output, a signature is added to the output data. To enable this feature, you can use the **utility sign** command along with the **show platform security integrity dossier** command. The output is presented in JSON format.

This **utility sign** can also be used with any of the IOS XR commands.

Note The Secure Unique Device Identifier or SUDI signature provides the hardware root of trust to the dossier that is collected by the system.

Verification Example of Collecting Data Dossier and Creating Signed-Envelope

```
Router#show  platform security integrity  dossier include  reboot-history nonce 1580 |
utility sign nonce 1580 include-certificate

Fri Mar 27 15:20:58.010 IST
{
 "cli-output":
"{\"collection-start-time\":1585302658.098761,\"model-name\":\"http://cisco.com/ns/yang/Cisco-IOS-XR-ma\",\"model-revision\":\"2019-08-05\",\"license-udi\":{\"result-code\":
 \"Success\", \"license-udi\": \"UDI:
PID:NCS-5501-SE,SN:FOC2107R0ZB\\n\"},\"version\":{\"result-code\": \"Success\", \"version\":
 \"Cisco IOS XR Software, Version 7.0.1.26I\\nCopyright (c) 2013-2020 by Cisco Systems,
Inc.\\n\\nBuild Information:\\n Built By     : labuser\\n Built On     : Wed Mar 11 20:46:36
 PDT 2020\\n
Built Host   : iox-ucs-009\\n Workspace    :
/auto/iox-ucs-009-san2/prod/7.0.1.26I.DT_IMAGE/asr9000/ws\\n Version      : 7.0.1.26I\\n
Location     : /opt/cisco/XR/packages/\\n Label        : 7.0.1.26I\\n\\ncisco ASR 9000 ()
processor\\nSystem uptime is 1 week 3 days 19 hours 58
minutes\\n\\n\"},\"platform\":{\"result-code\": \"Success\", \"platform\": \"Node
     Type                      State             Config
state\\n--------------------------------------------------------------------------------
--\\n0/RP0/CPU0        ASR-9000-SE(Active)        IOS XR RUN        NSHUT\\n0/RP0/NPU0
    Slice                     UP                \\n0/FT0             NCS-1RU-FAN-FW
       OPERATIONAL       NSHUT\\n0/FT1             NCS-1RU-FAN-FW             OPERATIONAL
      NSHUT\\n0/PM0             NCS-1100W-ACFW             FAILED            NSHUT\\n0/PM1
             NCS-1100W-ACFW            OPERATIONAL
NSHUT\\n\"},\"reboot-history\":{\"result-code\":\"Success\",\"model-name\":\"Cisco-IOS-XR-linu
x-os-reboot-history-oper\",\"model-revision\":\"2019-04-05\",\"node\":[{\"node-name\":
\"0/RP0/CPU0\", \"reboot-history\": [{\"reason\": \"User initiated graceful reload\",
\"time\": \"Wed Feb 19 15:25:11 2020\", \"cause-code\": 1, \"no\": 1}, {\"reason\":
\"CARD_SHUTDOWN\", \"time\": \"Wed Feb 19 16:38:00 2020\", \"cause-code\": 37, \"no\": 2},
 {\"reason\": \"CARD_SHUTDOWN\", \"time\": \"Wed Feb 19 19:06:27 2020\", \"cause-code\":
37, \"no\": 3}, {\"reason\": \"CARD_SHUTDOWN\", \"time\": \"Thu Feb 20 11:50
:50 2020\", \"cause-code\": 37, \"no\": 4}, {\"reason\": \"CARD_SHUTDOWN\", \"time\": \"Fri
 Feb 21 10:54:09 2020\", \"cause-code\": 37, \"no\": 5}, {\"reason\": \"CARD_SHUTDOWN\",
\"time\": \"Fri Feb 21 19:00:10 2020\", \"cause-code\": 37, \"no\": 6}, {\"reason\":
\"CARD_SHUTDOWN\", \"time\": \"Sun Feb 23 12:05:25 2020\", \"cause-code\": 37, \"no\": 7},
 {\"reason\": \"User initiated graceful reload\", \"time\": \"Mon Mar  2 19:03:25 2020\",
\"cause-code\": 241, \"no\": 8}, {\"reason\": \"CARD_SHUTDOWN\", \"
time\": \"Mon Mar  2 19:08:16 2020\", \"cause-code\": 37, \"no\":
9}]}]},\"collection-end-time\":1585302661.316119}",
 "signature-envelop": {
  "nonce": "1580",
  "signature-version": "01",
```

```
  "digest-algorithm": "RSA-SHA256",
  "pub-key-id": "4278",
  "signature":
"ZEKkhGKqZZifp3m6v/6O69MvXN+o9x+6vp9DnzO8YwaMdd59ORVRck9UoqWGd9JB9wfK9B7eMN+UvhCqBRwgw==",
  "sudi-signature":
"UogQoTKcJ5FFHQ3VYIBjYTelQax5b/I5yHcGL2xjw0HE27vtc7d2OQ7dC3rAljtkrlEZduAKHxhmkMoakR
Grp7gl+5PfsSeXdEMG3kaaKja3isPsyX2/EaBr3bw3SzUHaFicY3MPESS4FwdOpfVbEwe+AR+CB9lDnbl4Izwo0zDTw4M41SWkZZmgHVMXgVf
jwPiVYONdFVTift7rfoIoMVUoYkRbQYiFPGxMjgNcixfDqGjXoTt+hen4IRbvvRz653qgWVvS+TEgcU/nBVvkX1itNR5uGeh/Vcs8dbpBPixh
afZEfwWI8G2WQClfC0q+O+ggfn8ln9UW6exNKQZb2Q==",
"signing-certificate": [
"MIIDLjCCAhagAwIBAwICELUwDQYJKoZIhvcNAQELBQAwODEjMCEGCSqGSIb3DQEJAhYUdHVya
Z/tJlIYOzTRJjx9ZtFdX8yyOj3zuI+zDakPRn4XA2blqFN3dO71MofsIiO7SEKc52aQDes4PbjkQcibKYhrYboECypdhuG/TPyhxndFlWa/
ZnhGiziW7I9nddMgU5cE0XZ48x5G5ixqmwG8AQiuQHsNsCZ/hDeJiLrfOYYmlVXaRLZTJDZvuXqpmTn9k342NT+fqsHHvT+qyLZ5V9iuma
QyjHiP8I4kfVS5nzZhTjkEnQHgxadsNEY1pnThDntAEFsZacajHBFDNi1UyzbHxr0EwCc5ALpdyY1F9CghdcJ2XEd8VjGFtLXn1oFQvJe
Ru5e5BfM7+rU8IN3iuyLHAgMBAAGjZDBiMA8GA1UdEwEB/wQFMAMBAf8wDgYDVR0PAQH/BAQDAgP4MCAGA1UdJQEB/
eqW4mYYmDV+OE/BMszvjLl2wsIwDQYJKoZIhvcNAQELBQADggEBAHJL4re6ehAejTXBMGQAsIJ2Z4vdxeRb3N4qB1
EH3nUMxDmea5JCGO3b8=" ]
 }
}
Router#
```

Please note that the above output is a sample output which does not indicate actual values.

Collect Filesystem Inventory

The metadata of the filesystem can be collected using data dossier. The metadata of the file includes information about time the file was created, last accessed, last modified and so on. A snapshot is captured at each configured interval. The initial snapshot shows a complete snapshot of all files in the filesystem. The files are scanned periodically and new inventory data is collected and stored as incremental snapshots.

To enable this feature, use the **filesystem-inventory** command.

```
Router(config)#filesystem-inventory
Router(config-filesystem-inventory)#snapshot-interval 2
Router(config-filesystem-inventory)#commit
```

The `snapshot-interval` is the time interval in 15-minute blocks. The interval ranges 1–96. For example, value of `2` indicates that a snapshot interval is collected every 30 minutes. The snapshots are stored in./`misc/scratch/filesysinv` The logs are stored in `/var/log/iosxr/filesysinv/*`.

To retrieve the filesystem inventory, use the following dossier command. Output is presented in JSON format.

```
show platform security integrity dossier include filesystem-inventory | file
<platform>-parent.json

{"collection-start-time":1610168028.380901,
"model-name":"http://cisco.com/ns/yang/Cisco-IOS-XR-ama",
"model-revision":"2019-08-05","license-udi":{"result-code": "Success", "license-udi":
"UDI: PID:NCS-55A1-24H,SN:FOC2104R15R\n"},"version":{"result-code": "Success",
"version": "Cisco IOS XR Software, Version 7.3.1
\nCopyright (c) 2013-2020 by Cisco Systems, Inc.\n\nBuild Information:\n
Built By     : <user>\n Built On     : Thu Jan  7 17:16:02 PST 2021\n
Built Host   : <host>\n Workspace    : <ws>
Version     : 7.3.1\n  Location    : /opt/cisco/XR/packages/\n Label       : 7.3.1\n\ncisco

() processor\nSystem uptime is 8 hours 7 minutes\n\n"},"platform":{"result-code":
"Success", "platform":
"Node              Type                       State             Config state
--------------------------------------------------------------------------------
0/RP0/CPU0        <node-type>(Active)       IOS XR RUN        NSHUT\n
0/RP0/NPU0        Slice                      UP
0/RP0/NPU1        Slice                      UP
```

```
0/FT0              <platform>-A1-FAN-RV       OPERATIONAL       NSHUT
0/FT1              <platform>-A1-FAN-RV       OPERATIONAL       NSHUT
0/FT2              <platform>-A1-FAN-RV       OPERATIONAL       NSHUT
PM1                <platform>-1100W-ACRV      OPERATIONAL       NSHUT
"},
---------------------------Output is snipped for brevity
-----------------------------------
```

To limit the number of snapshots, use the following command:

```
show platform security integrity dossier include filesystem-inventory
filesystem-inventory-options '{\"0/RP0/CPU0\": {\"block_start\": 0, \"count\": 1}}'
```

To start from a new block, use the following command:

```
show platform security integrity dossier include filesystem-inventory
filesystem-inventory-options '{\"0/RP0/CPU0\": {\"block_start\": 5}}'
```

To collect data from a remote node, use the following command:

```
show platform security integrity dossier include filesystem-inventory
filesystem-inventory-options '{\"0/RP1/CPU0\": {\"block_start\": 0}}' | file
harddisk:PE1_remote.json
```

Following is the sample of the display compact container:

```
{"node-data":[{"node-location":"node0_RP0_CPU0","up-time":150311,"start-time":"Tue Jul 27
13:55:12 2021","ima-event-log-compact":
["IlYIABoMCO+ggIgGEKmxwZYBIkQIABAKGhTU2yPVDA5Rx+64ecp41qZQrLEWSCACKhSXl+340O7Ta
xz5JUeBYFHIr05F7jIOYm9vdF9hZ2dyZWdhdGVAAQ=="]}]}
```

Incremental Integrity Measurement Architecture

With incremental Integrity Meansurement Architecture (IMA), you can define the starting IMA sequence that you want to include in a response. The system then starts to report the subsequent events.

```
show platform security integrity dossier incremental-ima
"{\"ima_start\":[{\"0/RP0/CPU0\":{\"start_event\":1000,\"start_time\":\"Tue Feb 16 09:15:17
 2021\"}}]}"
```

Associated Command(s)

- **show platform security integrity dossier**
- **utility sign**

Procedure to Test Key Generation and Data-signing with Different Key Algorithm

You can follow these steps to test key generation and data-signing with a different key algorithm:

- Unconfigure the trustpoint (using the **no crypto ca trustpoint system-trustpoint** command)
- Clear the certificates that were generated earlier (using the **clear crypto ca certificates system-trustpoint** command)
- Generate new keys.
- Configure the system trustpoint again.
- Authenticate and enroll the system trustpoint to generate the certificates.

See How to Integrate Cisco IOS XR and Crosswork Trust Insights, on page 160 section for configuration steps of each task.

Verify Authenticity of RPM Packages Using Fingerprint

Table 24: Feature History Table

Feature Name	Release Information	Description
Verify Authenticity of RPM Packages Using Fingerprint	Release 7.3.1	This feature helps in verifying the authenticity of an installable package using fingerprint values. The fingerprint value of the package is compared with a point of reference called Known Good Value (KGV). The KGV for an image or package is generated after it is built by Cisco. After installing the package, the associated install time and build time fingerprint values are compared using Yang RPC to determine whether the package is genuine. A match in the fingerprints indicates that the package published on CCO and that installed on router are the same.

Is there a simple way to determine the authenticity of a package that is installed on a router? Is there a mechanism to identify whether a package signature is checked at install time, or detect changes to the files after the package is installed at run time?

Cisco IOS XR, Release 7.3.1 introduces a fingerprint mechanism to verify the authenticity of a package that Cisco releases. This mechanism helps determine whether the installed package is genuine, where the installed and running software matches the software that is published by Cisco.

There are significant security measures for installing software using GPG and IMA signing. However, there is need to report more data for Cisco Crosswork application to monitor and flag potential issues for further investigation. Cisco Crosswork monitors the installed software over a period to help accomplish the following tasks:

- To determine whether there are any differences between the software that is published on Cisco.com and that downloaded to the router.
- To determine whether any files in a package have been altered, either accidentally or maliciously, from the time the package was installed.

A Known Good Value (KGV) is calculated and published for each package. This value is considered the right value for the package.

Two fingerprint (hex) values for each active or committed packages are monitored to ensure authenticity of the package:

- **Install time fingerprint:** Hex value that represents the software in the package at install time. An RPM is genuine if it is not modified before install, and it matches the KGV. Whereas a manipulated RPM shows a mismatch in the fingerprint that is published in the KGV.
- **Run time fingerprint:** Hex value that represents the running software of an installed package. The value matches the corresponding install time fingerprint if the RPM has not been modified since the install time. If there are changes to the files, the run time and install time fingerprints show a mismatch. Every time the files that are installed by an RPM are changed, the run time fingerprint also changes. A value of `0` (zero) is displayed if no run time fingerprint is available for a package. This is used to monitor changes to the running software over time.



Note These two values are displayed only in the Yang model output. No CLI commands are provided to view these values.

```
Received message from host
<?xml version="1.0" ?>
<rpc-reply message-id="urn:uuid:97f5bc36-0eb0-4d2f-9c6f-3d34fea14be0"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
<data>
    <install xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-spirit-install-instmgr-oper">
      <packages>
        <active>
         <summary>
          <rpm-fingerprint-status>generation-up-to-date</rpm-fingerprint-status>
           <rpm-fingerprint-timestamp>Mon Jun 15 15:58:22 2020</rpm-fingerprint-timestamp>

            <package>
             <name>asr9k-xr</name>
              <version>7.3.1</version>
              <release>r731</release>
             <gpg-key-id>ddcead3dcb38048d</gpg-key-id>
              <rpm-fingerprint>

<rpm-fingerprint-install-time>2871bf68d3cd764938775afc9e5a69c130f9fbde</rpm-fingerprint-install-time>


<rpm-fingerprint-run-time>2871bf68d3cd764938775afc9e5a69c130f9fbde</rpm-fingerprint-run-time>

              </rpm-fingerprint>
            </package>

           <package>
            <name>asr9k-mcast-x64</name>
            <version>2.0.0.0</version>
            <release>r731</release>
            <gpg-key-id>ddcead3dcb38048d</gpg-key-id>
             <rpm-fingerprint>

<rpm-fingerprint-install-time>3ddca55bc00a0ce2c2e52277919d398621616b28</rpm-fingerprint-install-time>


<rpm-fingerprint-run-time>3ddca55bc00a0ce2c2e52277919d398621616b28</rpm-fingerprint-run-time>

              </rpm-fingerprint>
            </package>
------------------------------ Truncated for brevity ---------------------------------
```

In the example, both the install time and run time fingerprints are the same.

The fingerprint generation status is used to indicate how up-to-date the run time fingerprints are. This may indicate that generation is currently in progress and will complete shortly, or generation is awaiting the end of an atomic change.

Support for Ed25519 Public-Key Signature System

Table 25: Feature History Table

Feature Name	Release Information	Feature Description
Support for Ed25519 Public-Key Signature System	Release 7.3.1	This feature allows you to generate and securely store crypto key pair for the Ed25519 public-key signature algorithm on Cisco IOS XR 64-bit platforms. This signature system provides fast signing, fast key generation, fool proof session keys, collision resilience, and small signatures. The feature also facilitates integration of Cisco IOS XR with Cisco Crosswork Trust Insights. Commands introduced for this feature are: crypto key generate ed25519 crypto key zeroize ed25519 show crypto key mypubkey ed25519 Commands modified for this feature are: ca-keypair keypair

The Cisco IOS XR Software Release 7.3.1 introduces the support for Ed25519 public-key signature algorithm on 64-bit platforms. Prior to this release, only DSA, ECDSA, and RSA signature algorithms were supported. The Ed25519 signature algorithm uses the elliptic curve cryptography that offers a better security with faster performance when compared to other signature algorithms.

You can generate the Ed25519 crypto keys either with an empty label or with two predefined labels: **system-root-key** and **system-enroll-key**. In case of empty label, the system generates the key pair against the default label. You can use the key pairs with the predefined labels to integrate Cisco IOS XR with Cisco Crosswork Trust Insights.

Generate Crypto Key for Ed25519 Signature Algorithm

Configuration Example

To generate the Ed25519 crypto key, use the **crypto key generate ed25519** command in EXEC mode.

```
Router#crypto key generate ed25519
```

To delete the Ed25519 crypto key with default label or any predefined label, use the **crypto key zeroize ed25519** command in EXEC mode.

Verification

Use the **show crypto key mypubkey ed25519** command to view all Ed25519 crypto keys generated on the system.

```
Router# show crypto key mypubkey  ed25519

Mon Nov 30 07:05:06.532 UTC
Key label: the_default
Type : ED25519
Size : 256
Created : 07:03:17 UTC Mon Nov 30 2020
Data :
FF0ED4E7 71531B3D 9ED72C48 3F79EC59 9EFECCC3 46A129B2 FAAA12DD EE9D0351
```

Related Topics

- Support for Ed25519 Public-Key Signature System, on page 174
- Integrate Cisco IOS XR with Cisco Crosswork Trust Insights using Ed25519, on page 175

Associated Commands

- **crypto key generate ed25519**
- **crypto key zeroize ed25519**
- **show crypto key mypubkey ed25519**

Integrate Cisco IOS XR with Cisco Crosswork Trust Insights using Ed25519

Configuration Example

This section shows how to generate the system trustpoint, and the root and leaf certificates using the Ed25519 signature algorithm, as part of integrating Cisco IOS XR with Cisco Crosswork Trust Insights.

```
Router#configure
Router(config)#domain name domain1
Router(config)#crypto ca trustpoint system-trustpoint
Router(config-trustp)#keypair ed25519 system-enroll-key
Router(config-trustp)#ca-keypair ed25519 system-root-key
```

```
Router(config-trustp)#commit

/* Generate root and leaf certificates */
Router#crypto ca authenticate system-trustpoint
Router#crypto ca enroll system-trustpoint
```

Running Configuration

```
config
domain name domain1
crypto ca trustpoint system-trustpoint
 keypair ed25519 system-enroll-key
 ca-keypair ed25519 system-root-key
!
```

For the complete integration procedure, see, Integrating Cisco IOS XR and Crosswork Trust Insights, on page 159.

Where to Go Next

After you have finished configuring CA interoperability, you should configure IKE, IPSec, and SSL. IPSec in the *Implementing IPSec Network Security on* the Cisco ASR 9000 Series Routermodule, and SSL in the *Implementing Secure Socket Layer on* the Cisco ASR 9000 Series Routermodule. These modules are located in *System Security Configuration Guide for Cisco ASR 9000 Series Routers* (this publication).

Additional References

The following sections provide references related to implementing certification authority interoperability.

Related Documents

Related Topic	Document Title
PKI commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Public Key Infrastructure Commands on* the Cisco ASR 9000 Series Router module in *System Security Command Reference for Cisco ASR 9000 Series Routers*.

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

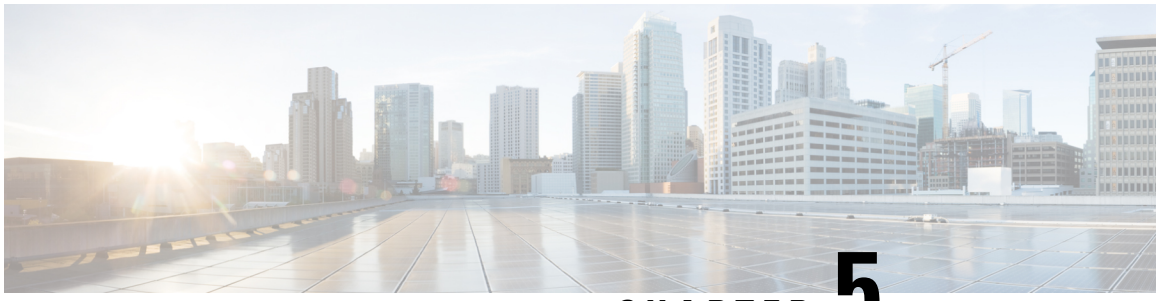
MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	—

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport



CHAPTER 5

Implementing Keychain Management

This module describes how to implement keychain management on. Keychain management is a common method of authentication to configure shared secrets on all entities that exchange secrets such as keys, before establishing trust with each other. Routing protocols and network management applications on Cisco IOS XR software often use authentication to enhance security while communicating with peers.

Feature History for Implementing Keychain Management

Release	Modification
Release 3.7.2	This feature was introduced.

- Prerequisites for Configuring Keychain Management, on page 179
- Restrictions for Implementing Keychain Management, on page 179
- Information About Implementing Keychain Management, on page 179
- How to Implement Keychain Management, on page 180
- Configuration Examples for Implementing Keychain Management, on page 190
- Additional References, on page 190

Prerequisites for Configuring Keychain Management

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Restrictions for Implementing Keychain Management

You must be aware that changing the system clock impacts the validity of the keys in the existing configuration.

Information About Implementing Keychain Management

The keychain by itself has no relevance; therefore, it must be used by an application that needs to communicate by using the keys (for authentication) with its peers. The keychain provides a secure mechanism to handle the keys and rollover based on the lifetime. Border Gateway Protocol (BGP), Open Shortest Path First (OSPF),

and Intermediate System-to-Intermediate System (IS-IS) use the keychain to implement a hitless key rollover for authentication. BGP uses TCP authentication, which enables the authentication option and sends the Message Authentication Code (MAC) based on the cryptographic algorithm configured for the keychain. For information about BGP, OSPF, and IS-IS keychain configurations, see *Routing Configuration Guide for Cisco ASR 9000 Series Routers*.

- Resource Reservation Protocol (RSVP) uses keychain for authentication. For more information about RSVP, see the *Cisco ASR 9000 Series Aggregation Services Router MPLS Configuration Guide*.
- IP Service Level Agreements (IP SLAs) use a keychain for MD5 authentication for the IP SLA control message. For more information about IP SLAs, see the *Cisco ASR 9000 Series Aggregation Services Router System Monitoring Configuration Guide* and the **key-chain** command in the *Cisco ASR 9000 Series Aggregation Services Router System Monitoring Comand Reference*.

To implement keychain management, you must understand the concept of key lifetime, which is explained in the next section.

Lifetime of Key

If you are using keys as the security method, you must specify the lifetime for the keys and change the keys on a regular basis when they expire. To maintain stability, each party must be able to store and use more than one key for an application at the same time. A keychain is a sequence of keys that are collectively managed for authenticating the same peer, peer group, or both.

Keychain management groups a sequence of keys together under a keychain and associates each key in the keychain with a lifetime.



Note Any key that is configured without a lifetime is considered invalid; therefore, the key is rejected during configuration.

The lifetime of a key is defined by the following options:

- Start-time—Specifies the absolute time.
- End-time—Specifies the absolute time that is relative to the start-time or infinite time.

Each key definition within the keychain must specify a time interval for which that key is activated; for example, lifetime. Then, during a given key's lifetime, routing update packets are sent with this activated key. Keys cannot be used during time periods for which they are not activated. Therefore, we recommend that for a given keychain, key activation times overlap to avoid any period of time for which no key is activated. If a time period occurs during which no key is activated, neighbor authentication cannot occur; therefore, routing updates can fail.

Multiple keychains can be specified.

How to Implement Keychain Management

This section contains the following procedures:

Configuring a Keychain

This task configures a name for the keychain.

You can create or modify the name of the keychain.

SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **commit**
4. **show key chain** *key-chain-name*
5. **show run**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**key chain** *key-chain-name*
	Example:
	`RP/0/RSP0/CPU0:router(config)# key chain isis-keys`
	`RP/0/RSP0/CPU0:router(config-isis-keys)#`
		Creates a name for the keychain.
		Note Configuring only the keychain name wit exit the configuration, the router does no identifier and at least one of the global c attributes (for example, lifetime or key s

Step 3	**commit**	Commits the configuration changes and remains within the configuration session.

Step 4	**show key chain** *key-chain-name*
	Example:
	`RP/0/RSP0/CPU0:router# show key chain isis-keys`
	`Key-chain: isis-keys/ -`
	`accept-tolerance -- infinite`
	`Key 8 -- text "1104000E120B520005282820"`
	`  cryptographic-algorithm -- MD5`
	`  Send lifetime:   01:00:00, 29 Jun 2006 - Always valid  [Valid now]`
	`  Accept lifetime: 01:00:00, 29 Jun 2006 - Always valid [Valid now]`
		(Optional) Displays the name of the keychain.
		Note The *key-chain-name* argument is optiona all the keychains are displayed.

Step 5	**show run**	

	Command or Action	Purpose

	Example:
	
	`key chain isis-keys`
	`accept-tolerance infinite`
	`key 8`
	`key-string mykey91abcd`
	`cryptographic-algorithm MD5`
	`send-lifetime 1:00:00 june 29 2006 infinite`
	`accept-lifetime 1:00:00 june 29 2006 infinite`
	`!`
	`!`
	`!`	

What to do next

After completing keychain configuration, see the Configuring a Tolerance Specification to Accept Keys, on page 182 section.

Configuring a Tolerance Specification to Accept Keys

This task configures the tolerance specification to accept keys for a keychain to facilitate a hitless key rollover for applications, such as routing and management protocols.

SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **accept-tolerance** *value* [**infinite**]
4. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**key chain** *key-chain-name* **Example:** `RP/0/RSP0/CPU0:router(config)# key chain isis-keys`	Creates a name for the keychain.

	Command or Action	Purpose
Step 3	**accept-tolerance** *value* [**infinite]** **Example:** `RP/0/RSP0/CPU0:router(config-isis-keys)# accept-tolerance infinite`	Configures an accept tolerance limit—duration for which an expired or soon-to-be activated keys can be used for validating received packets—for a key that is used by a peer. • Use the *value* argument to set the tolerance range in seconds. The range is from 1 to 8640000. • Use the **infinite** keyword to specify that an accept key is always acceptable and validated when used by a peer.
Step 4	**commit**	Commits the configuration changes and remains within the configuration session.

Configuring a Key Identifier for the Keychain

This task configures a key identifier for the keychain.

You can create or modify the key for the keychain.

SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*
4. **commit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**key chain** *key-chain-name* **Example:** `RP/0/RSP0/CPU0:router(config)# key chain isis-keys`	Creates a name for the keychain.
Step 3	**key** *key-id* **Example:**	Creates a key for the keychain. The key ID has to be unique within the specific keychain. • Use the *key-id* argument as a 48-bit integer.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-isis-keys)# key 8`	
Step 4	**commit**	Commits the configuration changes and remains within the configuration session.

What to do next

After configuring a key identifier for the keychain, see the Configuring the Text for the Key String, on page 184 section.

Configuring the Text for the Key String

This task configures the text for the key string.

SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*
4. **key-string** [**clear** | **password**] *key-string-text*
5. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**key chain** *key-chain-name* **Example:** `RP/0/RSP0/CPU0:router(config)# key chain isis-keys`	Creates a name for the keychain.

Step 3	**key** *key-id*
	Example:
	`RP/0/RSP0/CPU0:router(config-isis-keys)# key 8`
	`RP/0/RSP0/CPU0:router(config-isis-keys-0x8)#`
		Creates a key for the keychain.

Step 4	**key-string** [**clear**	**password**] *key-string-text* **Example:**	Specifies the text string for the key.

	Command or Action	Purpose

	`RP/0/RSP0/CPU0:router(config-isis-keys-0x8)# key-string password 8`	• Use the **clear** keyword to specify the key string in clear text form; use the **password** keyword to specify the key in encrypted form.
		• For a string to be a valid password, it must comply with the following rules:
		  • It must contain an even number of characters.
		  • The minumum length is 4.
		  • The first two digits must be decimal numbers and the others must be hexadecimal numbers.
		  • The first two digits must not be greater than 53.
		Examples of valid passwords are:
		  • 12abcd
		  • 32986510
		•

Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

What to do next

After configuring the text for the key string, see the Configuring the Keys to Generate Authentication Digest for the Outbound Application Traffic, on page 187 section.

Determining the Valid Keys

This task determines the valid keys for local applications to authenticate the remote peers.

SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*
4. **accept-lifetime** *start-time* [**duration** *duration-value* | **infinite** | *end-time*]

5. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**key chain** *key-chain-name* **Example:** `RP/0/RSP0/CPU0:router(config)# key chain isis-keys`	Creates a a name for the keychain.

Step 3	**key** *key-id*
	Example:
	`RP/0/RSP0/CPU0:router(config-isis-keys)# key 8`
	`RP/0/RSP0/CPU0:router(config-isis-keys-0x8)#`
		Creates a key for the keychain.

Step 4	**accept-lifetime** *start-time* [**duration** *duration-value* | **infinite** | *end-time*] **Example:** `RP/0/RSP0/CPU0:router(config-isis-keys-0x8)# accept-lifetime 1:00:00 october 24 2005 infinite`	(Optional) Specifies the validity of the key lifetime in terms of clock time. You can specify the *start-time* and *end-time* in *hh:mm:ss month DD YYYY* format or *hh:mm:ss DD month YYYY* format.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring the Keys to Generate Authentication Digest for the Outbound Application Traffic

This task configures the keys to generate authentication digest for the outbound application traffic.

SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*
4. **send-lifetime** *start-time* [**duration** *duration-value* | **infinite** | *end-time*]
5. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**key chain** *key-chain-name* **Example:** `RP/0/RSP0/CPU0:router(config)# key chain isis-keys`	Creates a name for the keychain.

Step 3	**key** *key-id*
	Example:
	`RP/0/RSP0/CPU0:router(config-isis-keys)# key 8`
	`RP/0/RSP0/CPU0:router(config-isis-keys-0x8)#`
		Creates a key for the keychain.

Step 4	**send-lifetime** *start-time* [**duration** *duration-value* | **infinite** | *end-time*]
	Example:
	`RP/0/RSP0/CPU0:router(config-isis-keys-0x8)#`
	`send-lifetime 1:00:00 october 24 2005 infinite`
		(Optional) Specifies the set time period during which an authentication key on a keychain is valid to be sent. You can specify the validity of the key lifetime in terms of clock time.
		In addition, you can specify a start-time value and one of the following values:
		• **duration** keyword (seconds)
		• **infinite** keyword
		• *end-time* argument

	Command or Action	Purpose
		If you intend to set lifetimes on keys, Network Time Protocol (NTP) or some other time synchronization method is recommended.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring the Cryptographic Algorithm

This task allows the key chain configuration to accept the choice of the cryptographic algorithm.

From Cisco IOS XR Software Release 7.1.2 and later, you must follow the below guidelines while configuring the key chain. These are applicable only for FIPS mode (that is, when **crypto fips-mode** is configured).

- You must configure the session with a FIPS-approved cryptographic algorithm. A session configured with non-approved cryptographic algorithm for FIPS (such as, **MD5** and **HMAC-MD5**) does not work. This is applicable for OSPF, BGP, RSVP, ISIS, or any application using key chain with non-approved cryptographic algorithm.
- If you are using any **HMAC-SHA** algorithm for a session, then you must ensure that the configured *key-string* has a minimum length of 14 characters. Otherwise, the session goes down.

SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*
4. **cryptographic-algorithm** [**HMAC-MD5** | **HMAC-SHA1-12** | **HMAC-SHA1-20** | **MD5** | **SHA-1** | **AES-128-CMAC-96** | **HMAC-SHA-256** | **HMAC-SHA1-96**]
5. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router# configure`	

Step 2	**key chain** *key-chain-name*
	Example:
	`RP/0/RSP0/CPU0:router(config)# key chain isis-keys`
	`RP/0/RSP0/CPU0:router(config-isis-keys)#`
		Creates a name for the keychain.

Step 3	**key** *key-id*
	Example:
	`RP/0/RSP0/CPU0:router(config-isis-keys)# key 8`
	`RP/0/RSP0/CPU0:router(config-isis-keys-0x8)#`
		Creates a key for the keychain.

Step 4	**cryptographic-algorithm** [**HMAC-MD5** | **HMAC-SHA1-12** | **HMAC-SHA1-20** | **MD5** | **SHA-1** | **AES-128-CMAC-96** | **HMAC-SHA-256** | **HMAC-SHA1-96**] **Example:** `RP/0/RSP0/CPU0:router(config-isis-keys-0x8)# cryptographic-algorithm MD5`	Specifies the choice of the cryptographic algorithm. You can choose from the following list of algorithms: • HMAC-MD5 • HMAC-SHA1-12 • HMAC-SHA1-20 • MD5 • SHA-1 • HMAC-SHA-256 • HMAC-SHA1-96 • AES-128-CMAC-96 The routing protocols each support a different set of cryptographic algorithms: • Border Gateway Protocol (BGP) supports HMAC-MD5, HMAC-SHA1-12, HMAC-SHA1-96 and AES-128-CMAC-96. • Intermediate System-to-Intermediate System (IS-IS) supports HMAC-MD5, SHA-1, MD5, AES-128-CMAC-96, HMAC-SHA-256, HMAC-SHA1-12, HMAC-SHA1-20, and HMAC-SHA1-96. • Open Shortest Path First (OSPF) supports MD5, HMAC-MD5, HMAC-SHA-256, HMAC-SHA1-12, HMAC-SHA1-20, and HMAC-SHA1-96.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions:

	Command or Action	Purpose
		• **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuration Examples for Implementing Keychain Management

This section provides the following configuration example:

Configuring Keychain Management: Example

The following example shows how to configure keychain management:

```
configure
 key chain isis-keys
  accept-tolerance infinite
  key 8
   key-string mykey91abcd
   cryptographic-algorithm MD5
   send-lifetime 1:00:00 june 29 2006 infinite
   accept-lifetime 1:00:00 june 29 2006 infinite


Router#show key chain isis-keys

Key-chain: isis-keys/ -

accept-tolerance -- infinite
Key 8 -- text "1104000E120B520005282820"
  cryptographic-algorithm -- MD5
  Send lifetime:   01:00:00, 29 Jun 2006 - Always valid  [Valid now]
  Accept lifetime: 01:00:00, 29 Jun 2006 - Always valid [Valid now]
```

Additional References

The following sections provide references related to implementing keychain management.

Related Documents

Related Topic	Document Title
Keychain management commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Keychain Management Commands in the System Security Command Reference for Cisco ASR 9000 Series Routers*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

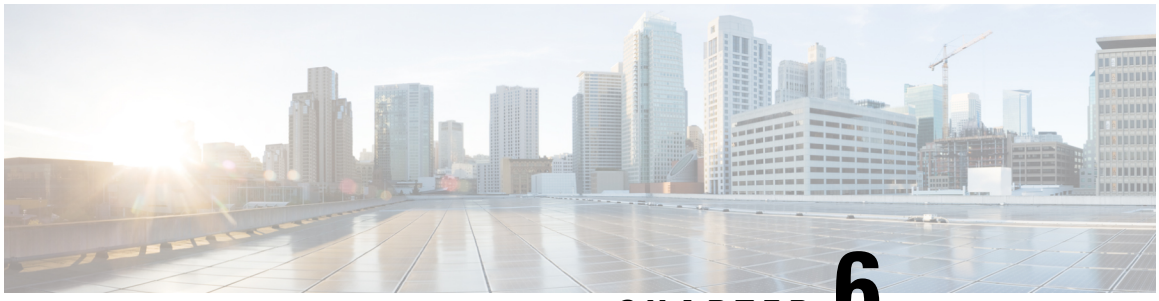
MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature.	—

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport



CHAPTER 6

Configure MACSec

This module describes how to configure Media Access Control Security (MACSec) encryption on the ASR 9000 Series Aggregation Services Routers. MACSec is a Layer 2 IEEE 802.1AE standard for encrypting packets between two MACSec-capable routers.

Feature History for Configure MACSec

Release	Modification
Release 5.3.2	This feature was introduced.
Release 6.0.1	This feature was modified to support VLAN sub-interfaces and bundles.
Release 6.1.2	This feature was modified to introduce MACsec as a service.
Release 6.3.3	Introduced the support for global MACsec shutdown.
Release 6.3.3	Introduced the support for MACsec SAK rekey interval.
Release 6.5.1	MACSec support was introduced on Cisco ASR 9901 Routers.
Release 6.6.1	A9K-MPA-32x1GE MPA card was introduced with MACSec support for Cisco IOS XR.
Release 6.6.2	MACSec support with A9K-MPA-32x1GE extended to IOS XR 64-bit.
Release 7.1.1	MACsec ISSU feature was introduced for 64-bit IOS XR.
Release 7.1.3	MACSec support was introduced on Cisco ASR 9000 5th generation line cards, Cisco ASR 9903 1.6T chassis and Cisco ASR 9903 2T port expansion card running Cisco IOS XR 64-bit.

- Understanding MACsec Encryption, on page 194
- Advantages of Using MACsec Encryption, on page 195
- Types of MACsec Implementation, on page 195
- Configure the EAP-derived MACsec CAK length, on page 196
- MKA Authentication Process, on page 197
- Hardware Support for MACSec, on page 198
- MACSec Limitations for Cisco ASR 9901 Routers, on page 200
- MACsec PSK, on page 201
- PQC key exchange groups for EAP-TLS, on page 201

- Fallback PSK, on page 205
- WAN MACsec, on page 206
- Configuring and Verifying MACSec Encryption , on page 209
- Configuring and Verifying MACsec Encryption as a Service, on page 246
- Quantum safe key distribution options for MACsec, on page 270
- Understanding SKIP, on page 278
- Global MACsec Shutdown, on page 284
- MACsec ISSU, on page 286
- MACsec SNMP MIB (IEEE8021-SECY-MIB), on page 292

Understanding MACsec Encryption

Security breaches can occur at any layer of the OSI model. At Layer 2, some of the common breaches at Layer 2 are MAC address spoofing, ARP spoofing, Denial of Service (DoS) attacks against a DHCP server, and VLAN hopping.

MACsec secures data on physical media, making it impossible for data to be compromised at higher layers. As a result, MACsec encryption takes priority over any other encryption method such as IPsec and SSL, at higher layers. MACsec is configured on Customer Edge (CE) router interfaces that connect to Provider Edge (PE) routers and on all the provider router interfaces.

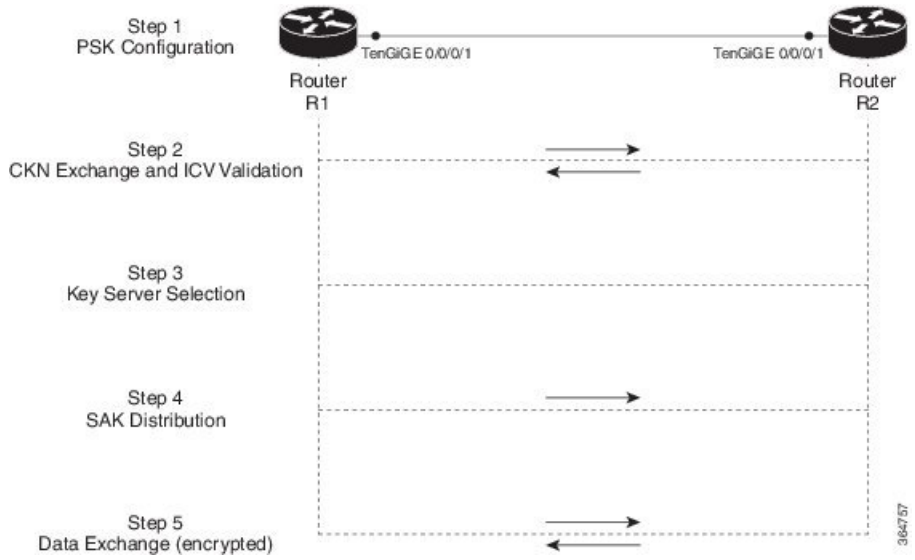
MACservice can be deployed in the network as a technology or as a service. For more information, see Types of MACsec Implementation, on page 195

MACsec Authentication Process

MACsec provides encryption using Advanced Encryption Standard (AES) algorithm at the Layer 2. MACsec uses the MACsec Key Agreement protocol (MKA) to exchange session keys, and manage encryption keys.

The MACsec encryption process is illustrated in the following figure and description.

Figure 6: MACsec Encryption Process



Step 1: When a link is first established between two routers, they become peers. Mutual peer authentication takes place by configuring a Pre-shared Key (PSK).

Step 2: On successful peer authentication, a connectivity association is formed between the peers, and a secure Connectivity Association Key Name (CKN) is exchanged. After the exchange, the MKA ICV is validated with a Connectivity Association Key (CAK), which is effectively a secret key.

Step 3: A key server is selected between the routers, based on the configured key server priority. Lower the priority value, higher the preference for the router to become the key server. If no value is configured, the default value of 16 is taken to be the key server priority value for the router. Lowest priority value configures that router as the key server, while the other router functions as a key client. The following rules apply to key server selection:

- Numerically lower values of key server priority and SCI are accorded the highest preference.
- Each router selects a peer advertising the highest preference as its key server provided that peer has not selected another router as its key server or is not willing to function as the key server.
- In the event of a tie for highest preferred key server, the router with the highest priority SCI is chosen as key server (KS).

Step 4: A security association is formed between the peers. The key server generates and distributes the Secure Association Key (SAK) to the key client (peer). SAKs are generated for every data exchange between the peers.

Step 5: Encrypted data is exchanged between the peers.

Advantages of Using MACsec Encryption

- **Client-Oriented Mode**: MACsec is used in setups where two routers that are peering with each other can alternate as a key server or a key client prior to exchanging keys. The key server generates and maintains the CAK between the two peers.
- **Data Integrity Check**: MACsec uses MKA to generate an Integrity Check Value (ICV) for the frame arriving on the port. If the generated ICV is the same as the ICV in the frame, then the frame is accepted; otherwise it is dropped.
- **Data Encryption**: MACsec provides port-level encryption on the line card of the router. This means that the frames sent out of the configured port are encrypted and frames received on the port are decrypted. MACsec also provides a mechanism where you can configure whether only encrypted frames or all frames (encrypted and plain) are accepted on the interface.
- **Replay Protection**: When frames are transmitted through the network, there is a strong possibility of frames getting out of the ordered sequence. MACsec provides a configurable window that accepts a specified number of out-of-sequence frames.
- **Support for Clear Traffic**: If configured accordingly, data that is not encrypted is allowed to transit through the port.

Types of MACsec Implementation

MACsec is implemented in the following ways:

- **MACsec** where it serves as an encryption method for all traffic on Ethernet links.

For more information on configuring MACsec, see Configuring and Verifying MACSec Encryption , on page 209.

For insights into deployment scenarios, see WAN MACsec, on page 206 .

- **MACsec as a service** where it serves as an encryption method for L2VPN and L3VPN traffic over a provider network. It provides a mechanism to provide encryption or decryption service for selected traffic across the WAN core. For example: aservice provider can charge encryption of voice calls at a premium. This solution supports both Point-to-Point as well as Multipoint service for all the traffic on the network.

  For more information on configuring MACsec as a service, see Configuring MACsec as a Service, on page 248

Both MACsec and MACsec service are mutually exclusive and can be deployed in the same network.

Configure the EAP-derived MACsec CAK length

Set the 128-bit Connectivity Association Key (CAK) length used for MACsec EAP sessions.

MACsec EAP supports configuring the EAP-derived CAK length as either 128 bits or 256 bits under a MACsec policy. The default EAP-derived CAK length is 128 bits.

Before you begin

Identify the MACsec policy used by the MACsec EAP session.

Follow these steps to configure the EAP-derived MACsec CAK length:

Procedure

Step 1 Configure the EAP-derived CAK length in the MACsec policy.

Example:

```
Router(config)# macsec-policy <policy-name> eap-cak-length 128
Router(config)# commit
```

Step 2 Display and verify the configured EAP-derived CAK length.

Example:

```
Router# show mka macsec policy <policy-name>
```

Changing the EAP-derived CAK length in a MACsec policy re-establishes MACsec sessions on all interfaces that use the policy with the newly configured CAK length.

What to do next

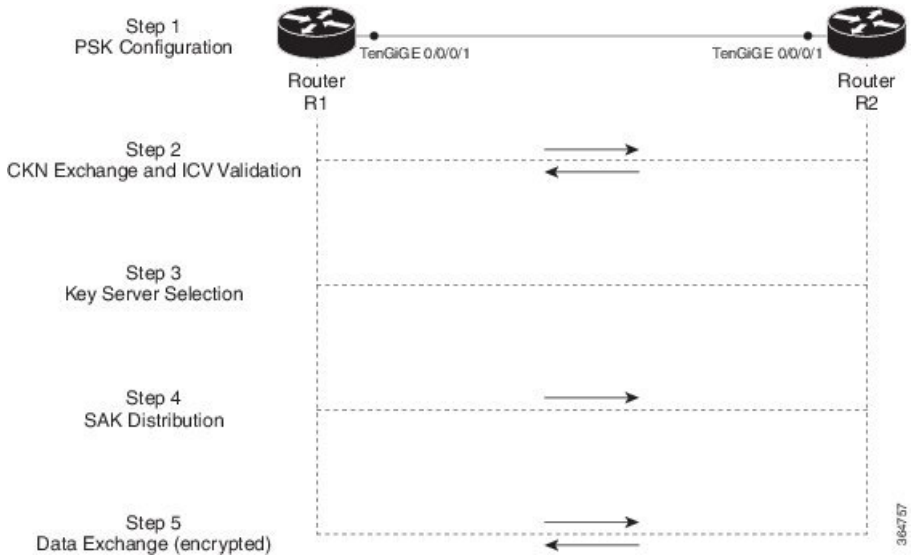
Configure the same EAP-derived CAK length in the MACsec policy on the peer.

MKA Authentication Process

MACsec provides encryption at the Layer 2, which is provided by the Advanced Encryption Standard (AES) algorithm that replaces the DES algorithm. MACsec uses the MACsec Key Agreement protocol (MKA) to exchange session keys, and manage encryption keys.

The MACsec encryption process is illustrated in the following figure and description.

Figure 7: MKA Encryption Process



Step 1: When a link is first established between two routers, they become peers. Mutual peer authentication takes place by configuring a Pre-shared Key (PSK).

Step 2: On successful peer authentication, a connectivity association is formed between the peers, and a secure Connectivity Association Key Name (CKN) is exchanged. After the exchange, the MKA ICV is validated with a Connectivity Association Key (CAK), which is effectively a secret key.

Step 3: A key server is selected between the routers, based on the configured key server priority. Lower the priority value, higher the preference for the router to become the key server. If no value is configured, the default value of 16 is taken to be the key server priority value for the router. Lowest priority value configures that router as the key server, while the other router functions as a key client. The following rules apply to key server selection:

- Numerically lower values of key server priority and SCI are accorded the highest preference.
- Each router selects a peer advertising the highest preference as its key server provided that peer has not selected another router as its key server or is not willing to function as the key server.
- In the event of a tie for highest preferred key server, the router with the highest priority SCI is chosen as key server (KS).

Step 4: A security association is formed between the peers. The key server generates and distributes the Secure Association Key (SAK) to the key client (peer). Each secure channel is supported by an overlapped sequence of Security Associations(SA). Each SA uses a new Secure Association Key (SAK).

Step 5: Encrypted data is exchanged between the peers.

MACsec Frame Format

The MACsec header in a frame consists of three components as illustrated in the following figure.

- **Security tag**: The security tag is 8-16 bytes in length and identifies the SAK to be used for the frame. With Secure Channel Identifier (SCI) encoding, the security tag is 16 bytes in length, and without the encoding, 8 bytes in length (SCI encoding is optional).The security tag also provides replay protection when frames are received out of sequence.
- **Secure data**: This is the data in the frame that is encrypted using MACsec and can be 2 or more octets in length.
- **ICV**: The ICV provides the integrity check for the frame and is usually 8-16 bytes in length, depending on the cipher suite. Frames that do not match the expected ICV are dropped at the port.

Figure 8: MACsec Frame Format



Hardware Support for MACSec

The MACSec support on ASR 9000 Series Routers is compatible with the following chassis, line cards (LCs), and modular port adapters (MPAs).

Cisco IOS XR Software Release 7.3.2 and Release 7.4.1 introduce MACSec on sub-interfaces of ASR 9000 5th Generation Line Cards. For detailed list of supported PIDs, see the section, *Supported Line Cards for MACSec*.

Supported Chassis for MACSec

Table 26: Supported Chassis for MACSec

Chassis Type	Introduced Release for MACSec Support
Cisco ASR 9903 Router (with removable A9903-8HG-PEC card)	Release 7.4.1
Cisco ASR 9902 Router	Release 7.4.1
Cisco ASR 9903 Router (1.6T Fixed Board only or with removable A9903-20HG-PEC card)	Release 7.1.3
Cisco ASR 9901 Router	Release 6.5.1

Supported Modular Port Adapters for MACSec

The MACSec technology is supported on modular line cards when used with the following MPAs:

Table 27: Supported MPAs for MACSec

Hardware PIDs	Hardware Description	Introduced Release for MACSec Support
A9K-MPA-32X1GE	32-port GE Modular Port Adapter	Release 6.6.1
A9K-MPA-20X10GE	20-port 10 Gigabit Modular Port Adapter	Release 6.1.2
A9K-MPA-1X100GE	1-port 100 Gigabit Modular Port Adapter	Release 6.1.2
A9K-MPA-2X100GE	2-port 100 Gigabit Modular Port Adapter	Release 6.1.2

Supported Line Cards and Port Expansion Cards for MACSec

Following line cards and port expansion cards support MACSec:

Table 28: Supported Line Cards for MACSec

Line Card	Introduced Release for MACSec Support
200G and 400G modular line cards with A9K-MPA-20X10GE, A9K-MPA-1X100GE and A9K-MPA-2X100GE	Release 6.1.2
200G and 400G modular line cards with A9K-MPA-32X1GE	Release 6.6.1
4X100 GE and 8X100 GE OTN Line Card	Release 6.1.2
Cisco ASR 9000 Series 400-Gbps IPoDWDM Line Card - A9K-400G-DWDM-TR	Release 6.2.1
ASR 9000 5th Generation Line Cards	*See the table below for the list of supported PIDs and release information*

Table 29: Supported Port Expansion Cards for MACSec

Hardware PID	Hardware Description	Introduced Release for MACSec Support (on main interface)	Introduced Release for MACSec Support (on sub-interface)
A9903-8HG-PEC	ASR 9903 800G Multirate Port Expansion Card	Release 7.4.1	Release 7.4.1
A9903-20HG-PEC	ASR 9903 2T Multirate Port Expansion Card	Release 7.1.3	Release 7.3.2

Table 30: Supported ASR 9000 5th Generation Line Cards for MACSec

Hardware PID	Hardware Description	Introduced Release for MACSec Support (on main interface)	Introduced Release for MACSec Support (on sub-interface)
A9K-4HG-FLEX-X-SE	ASR 9000 400GE Service Edge Line Card Enhanced - 5th Generation	Release 25.2.1	Release 25.2.1
A99-4HG-FLEX-SE	ASR 9900 400GE Combo Service Edge Line Card - 5th Generation	Release 7.4.1	Release 7.4.1
A99-4HG-FLEX-TR	ASR 9900 400GE Combo Packet Transport Line Card - 5th Generation	Release 7.4.1	Release 7.4.1
A99-10X400GE-X-SE	ASR 9000 4T Service Edge Line Card - 5th Generation	Release 7.3.1	Release 7.3.2
A99-10X400GE-X-TR	ASR 9000 4T Packet Transport Line Card - 5th Generation	Release 7.3.1	Release 7.3.2
A9K-20HG-FLEX-SE	ASR 9000 2T Service Edge Combo Line Card - 5th Generation	Release 7.1.3	Release 7.3.2
A9K-20HG-FLEX-TR	ASR 9000 2T Packet Transport Combo Line Card - 5th Generation	Release 7.1.3	Release 7.3.2
A9K-8HG-FLEX-SE	ASR 9000 800G Service Edge Combo Line Card - 5th Generation	Release 7.1.3	Release 7.3.2
A9K-8HG-FLEX-TR	ASR 9000 800G Packet Transport Combo Line Card - 5th Generation	Release 7.1.3	Release 7.3.2

Note

- MACSec is not supported on ASR9000 24-port dual-rate 10G/1G service edge–optimized line card (A9K-24X10GE-1G-SE).

MACSec Limitations for Cisco ASR 9901 Routers

The following MACSec limitations are applicable for Cisco ASR 9901 routers:

- 1 Gigabit Ethernet interface supports MACSec only for GCM-AES-128 cipher.
- 1 Gigabit Ethernet interfaces created from 24 multi-rate ports do not support MACSec.
- MACSec on VLAN is not supported.
- Point-to-Multipoint scenarios are not supported.
- MACSec as a service is not supported.

MACsec PSK

A pre-shared key includes a connectivity association key name (CKN) and a connectivity association key (CAK). A pre-shared key is exchanged between two devices at each end of a point-to-point (P2P) link to enable MACsec using static CAK security mode. The MACsec Key Agreement (MKA) protocol is enabled after the pre-shared keys are successfully verified and exchanged. The pre-shared keys, the CKN and CAK, must match on both ends of a link.

PQC key exchange groups for EAP-TLS

A PQC key exchange group is a TLS named group that

- combines classical and post-quantum cryptography (PQC) algorithms to protect key exchange,
- is advertised by IOS XR during 802.1X EAP-TLS negotiation, and
- enables support for hybrid, pure PQC, and classical key exchange methods for dynamic MACsec sessions.

Table 31: Feature History Table

Feature Name	Release Information	Feature Description
PQC-enabled MACsec with EAP-TLS	Release 26.3.1	Introduces PQC-capable TLS key exchange support to 802.1X EAP-TLS, with default CiscoSSL group advertisement and per-profile group configuration. This feature is supported on all MACsec-supported IOS XR platforms.

From IOS XR Release 26.3.1, EAP-TLS advertises CiscoSSL-supported PQC and classical groups by default when no explicit group list is configured. You can configure a colon-separated list on each EAP profile.

The feature supports the 802.1X EAP-TLS supplicant and the Local EAP authenticator. After successful authentication, the EAP-TLS session provides keying material for MACsec key establishment.

Note

- Hybrid groups are recommended during migration because they provide classical and PQC key exchange protection.
- On platforms that support port control, MACsec EAP does not support the `should-secure` MACsec security policy. Dot1x continues to perform port control as part of its default operation, which prevents `should-secure` from functioning with MACsec EAP.

Ensure compatibility and FIPS compliance for PQC EAP-TLS configuration

Follow these requirements when configuring PQC EAP-TLS key exchange groups:

- Ensure that the supplicant and authenticator have at least one key exchange group in common.
- If the peers have no group in common, EAP-TLS fails and the system logs an authentication or supplicant TLS failure with a no-common-group hint.
- Use only FIPS-approved groups when the system operates in FIPS mode.
- The system retains valid non-FIPS configured groups but excludes them from EAP-TLS authentication while FIPS mode is active.
- If FIPS excludes every configured group, the TLS session fails explicitly without using the default group list.
- Changing the configured key exchange groups restarts affected EAP-TLS and MACsec sessions.

Supported PQC and classical key exchange groups

Use this reference to select supported TLS key exchange groups for an EAP profile.

Table 32: Supported key exchange groups

Category	Groups	FIPS mode
Hybrid ML-KEM and ECDHE	x448_mlkem768, X25519MLKEM768, x25519_mlkem512, p256_mlkem512, p384_mlkem768, SecP256r1MLKEM768, p521_mlkem1024, SecP384r1MLKEM1024	p256_mlkem512, p384_mlkem768, SecP256r1MLKEM768, p521_mlkem1024, SecP384r1MLKEM1024
Pure ML-KEM	mlkem512, mlkem768, mlkem1024	Not permitted
Classical ECDHE	x25519, x448, secp256r1, secp384r1, secp521r1	secp256r1, secp384r1, secp521r1
Classical FFDHE	ffdhe2048, ffdhe3072, ffdhe4096	ffdhe2048, ffdhe3072, ffdhe4096

Configure PQC key exchange groups for EAP-TLS

Configure the key exchange groups that an EAP-TLS profile uses for dynamic MACsec authentication.

When no group list is configured, IOS XR uses the default group list provided by CiscoSSL. In FIPS mode, IOS XR considers only FIPS-approved groups during EAP-TLS negotiation. If a group list is configured, non-FIPS groups remain in the configuration but are not used while FIPS mode is active.

Note FIPS mode is enabled systemwide. If the router operates in FIPS mode, use only FIPS-approved key-exchange groups. For FIPS mode prerequisites, enablement, and operational guidance, refer to Configuring FIPS Mode.

Before you begin

- Identify the TLS trustpoint to be used.
- Ensure you have determined a group list that both EAP-TLS peers support.

Follow these steps to configure PQC key exchange groups for EAP-TLS:

Procedure

Step 1 Enter EAP profile TLS configuration mode.

Example:

```
Router(config)# eap profile <profile-name>
method tls
pki-trustpoint <trustpoint>
```

Step 2 Associate the EAP profile with the Dot1x profile.

Example:

```
Router(config)# dot1x
profile <dot1x-profile-name>
eap profile <eap-profile-name>
```

Step 3 Configure the colon-separated key exchange group list.

Example:

```
Router(config)# key-exchange-groups <colon-separated-list>
```

Step 4 To verify your configuration, display detailed Dot1x information for the interface to check TLS version, cipher suite, key exchange type, and group.

Example:

```
Router# show dot1x interface <interface-name> detail
```

Step 5 Display the configured groups in the EAP profile.

Example:

```
Router# show running-config eap profile <profile-name>
```

Changing the configured key exchange groups resets affected EAP-TLS and MACsec sessions and starts reauthentication.

What to do next

Configure the same or an overlapping group list on the EAP-TLS peer to ensure successful negotiation.

Troubleshoot EAP-TLS key exchange group negotiation

Identify and resolve mismatches in negotiated TLS key exchange groups causing EAP-TLS authentication failures.

When EAP-TLS authentication fails, IOS XR provides session-specific TLS failure hints in Dot1x output and may display FIPS warnings if FIPS mode excludes a configured key exchange group.

Before you begin

- Identify the affected interface.
- Collect the configured EAP profile and MACsec policy names.

Follow these steps to troubleshoot EAP-TLS key exchange group:

Procedure

Step 1 Check for `DOT1X-3-AUTH_TLS_FAILURE` or `DOT1X-3-SUPP_TLS_FAILURE` and their TLS failure hints. When FIPS excludes a configured non-FIPS group, check for `DOT1X-4-FIPS_WARN`.

Step 2 Display detailed Dot1x information for the affected interface.

Example:

```
Router# show dot1x interface <interface-name> detail
```

Check the TLS Fail Info field for a no-common-key-exchange-group hint or another session-specific TLS failure reason.

Step 3 Display the configured key exchange group list.

Example:

```
Router# show running-config eap profile <profile-name>
```

Compare the configured list with the group list on the peer and confirm that the lists overlap.

Step 4 Enable EAP method debugging when additional TLS handshake detail.

Example:

```
Router# debug eap method
```

The debug output reports ClientHello supported groups and versions, proposed cipher suites, and ServerHello selected values.

The diagnostic output identifies the TLS parameters that the peers proposed or selected and indicates whether configuration mismatch or FIPS filtering caused the failure.

Fallback PSK

Fallback is a session recovery mechanism when primary PSK fails to bring up secured MKA session. It ensures that a PSK is always available to perform MACsec encryption and decryption.

- In CAK rollover of primary keys, if latest active keys are mismatched, system performs a hitless rollover from current active key to fallback key, provided the fallback keys match.
- If a session is up with fallback, and primary latest active key configuration mismatches are rectified between peers, system performs a hitless rollover from fallback to primary latest active key.



Note

- A valid Fallback PSK (CKN and CAK) must be configured with infinite lifetime. If the fallback PSK is configured with CAK mismatch, the only recovery mechanism is to push a new set of PSK configurations (both on fallback PSK keychain and primary PSK chain in that order) on all the association members.
- In P2P topologies, a rollover to the fallback PSK happens when either of the nodes in the Secure Association (SA) cannot peer up with the primary PSK. Whereas, in P2MP, the fallback happens only at the expiry or deletion of the primary key on all peers, not just on one of the peers. On deletion or expiry of the primary PSK on one of the nodes, say R1, a new key server is chosen among the peer nodes that does a SAK rekey for the remaining nodes. This ensures that R1 is no longer part of the SA, and the network drops all traffic to and from R1.

The following is a sample syslog for session secured with fallback PSK:

```
%L2-MKA-5-SESSION_SECURED_WITH_FALLBACK_PSK : (Hu0/1/0/0) MKA session secured, CKN:ABCD
```

For more information on MACsec fallback PSK configuration, see Applying MACsec Configuration on an Interface, on page 217.

Active Fallback

The Cisco IOS XR Software Release 7.1.2 introduces the support for active fallback feature that initiates a fallback MKA session on having fallback configuration under the interface.

The key benefits of active fallback feature are:

- Faster session convergence on fallback, in the event of primary key deletion, expiry or mismatch.
- Faster traffic recovery under should-secure security policy when both primary and fallback mismatch happens.

With the introduction of active fallback functionality, the output of various MACsec show commands include the fallback PSK entry as well. If the session is secured with primary key, the fallback session will be in ACTIVE state. See, Verifying MACsec Encryption on IOS XR, on page 229 for details and sample outputs.



Note If the peer device is running on an older release that does not support active fallback feature, you must configure the **enable-legacy-fallback** command under the macsec-policy to ensure backward compatibility.

WAN MACsec

MACsec services over the WAN or Metro Ethernet offers Layer 2 transparent services such as E-Line or E-LAN using various transport layer protocols such as Ethernet over Multiprotocol Label Switching (EoMPLS).

WAN MACsec Use Cases

This section details the WAN MACsec use cases:

Use Case 1: MACSec in a L2VPN

The following figure illustrates the use of MACSec in a L2VPN network. In this topology, MACSec is configured on the PE-facing interfaces of the CE routers. The interfaces can be physical ethernet interfaces or VLAN sub-interfaces.

In a L2VPN network that uses an Ethernet over MPLS (EoMPLS) pseudowire, the traffic between CE routers is encrypted by MACSec with VLAN tags in clear. The following figure illustrates the use of MACSec in a L2VPN cloud using an EoMPLS pseudowire. MACSec is configured on the PE-facing VLAN sub-interfaces of the CE router. The PE router encapsulates the MACSec frames with VLAN tags and MPLS labels in clear and sends the frames over the EoMPLS pseudowire.

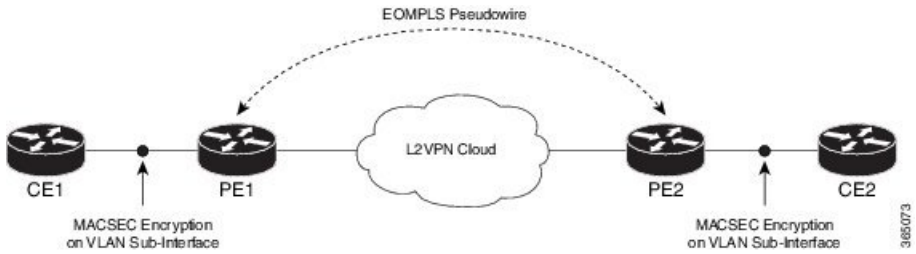
The following table lists the number of sub-interfaces with MACSec supported in a L2VPN.

Note To achieve scaling, sub-interfaces must be used.

Table 33: Supported MACSec Sessions on Sub-Interfaces

Interface Type	No. of Supported MACSec sessions (P2P)
10-GigE	5
40-GigE	21
100-GigE	42

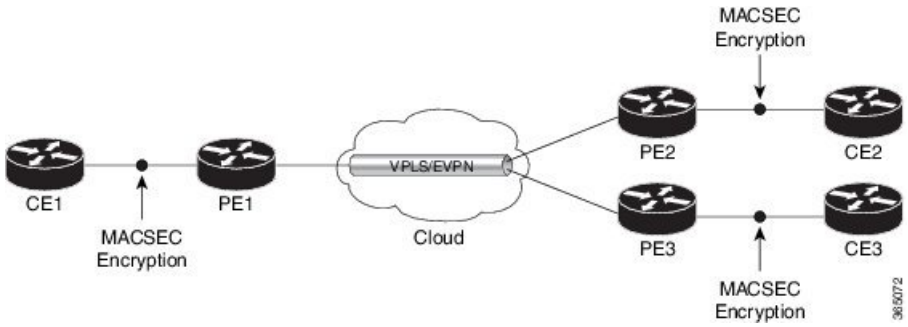
Figure 9: MACSec in a L2VPN Cloud



Use Case 2: MACSec in a VPLS/EVPN

A typical VPLS network often suffers the injection of labeled traffic from potential hackers. The following figure illustrates the use of MACSec in a VPLS/EVPN network for encrypting the data being exchanged over the VPLS cloud. In this topology MACSec is configured on the PE-facing interfaces of the CE routers. The interfaces can be physical ethernet interfaces or VLAN sub-interfaces.

Figure 10: MACSec in a VPLS/EVPN Cloud



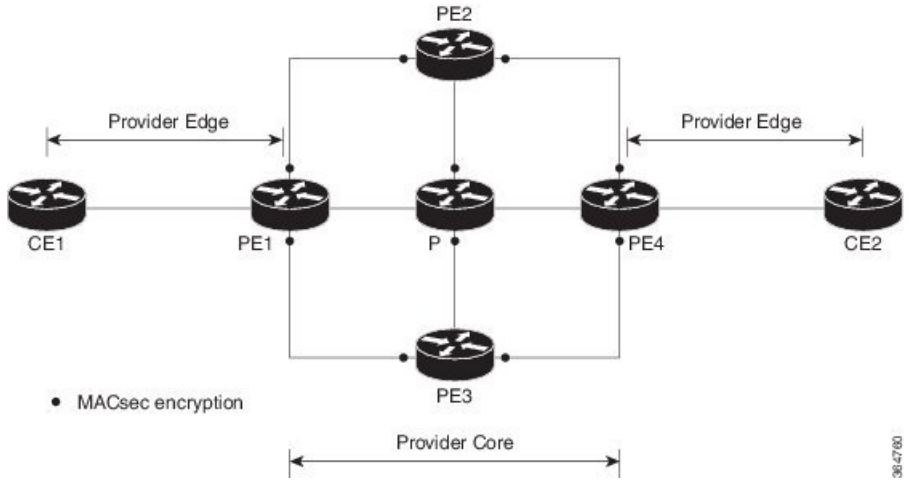
Use Case 3: MACSec in an MPLS Core Network

MACSec in an MPLS core network can be configured on physical interfaces, sub-interfaces or link bundles (Link Aggregation Group or LAG).

In the following topology, MACSec is configured on all router links in the MPLS core. This deployment is useful when the MPLS network spans data centers that are not co-located in the same geography. Each link is, therefore, a link between two data centers and all data exchanged is encrypted using MACSec.

The following figure illustrates the use of MACSec on physical interfaces in an MPLS core network.

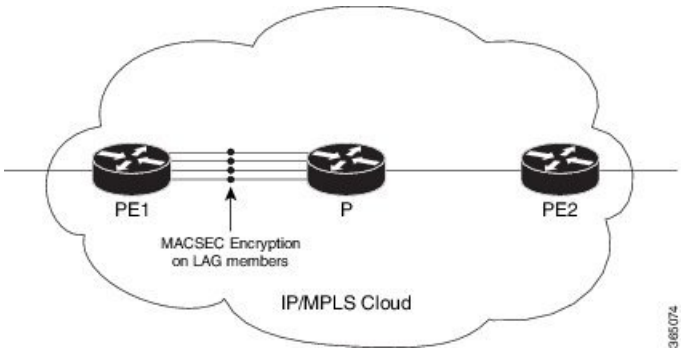
Figure 11: MACSec on Physical Interfaces in an MPLS Core Network



When MACSec is configured on the members of a LAG, an MKA session is set up for each member. SAK is exchanged for each LAG member and encryption/decryption takes place independently of other members in the group. MACSec can also be configured on VLAN sub-interfaces in these networks.

The following figure illustrates the use of MACSec on a link bundle in an MPLS core network.

Figure 12: MACSec on a Link Bundle in an MPLS Core Network



MACsec Encryption on Layer 3 Subinterface

You can now implement MACsec on L3 subinterfaces to provide secure communication within a specific L3 VLAN. On implementing MACsec on the L3 subinterface, the MACsec encryption and authentication are unique to the traffic on that subinterface. As a result, you can control the traffic encryption for individual subinterfaces of a physical interface by customizing MACsec policies.

MACsec on L3 subinterface configurations are similar to the MACsec configurations on a physical interface. For a successful MACsec Key Agreement protocol (MKA) session to be up on any L3 subinterface, it must have a valid tagging protocol encapsulation and a VLAN identifier assigned. All L3 subinterfaces always default to the 802.1Q VLAN encapsulation. However, the VLAN identifier must be explicitly defined.

To configure MACsec Encryption on Layer 3 Subinterface, refer Configuring and Verifying MACsec Encryption on VLAN Subinterfaces , on page 220.

Guidelines and Restrictions for MACsec Encryption on Layer 3 Subinterface

- MACsec Encryption on the Layer 3 Subinterface is supported only in the N540-24Q8L2DD-SYS router.
- The L3 subinterfaces belonging to a physical interface must have either of the following encapsulation combinations:
  - 802.1Q with a single tag
  - 802.1Q with double tags
  - 802.1ad with a single tag
  - 802.1ad with double tags
- You must configure the same type of VLAN tag on all the subinterfaces belonging to a physical interface.
- The MACsec encryption on layer 3 subinterface supports VLAN identifier range of 1–4094.
- The encapsulation configured on the L3 subinterface and the number of VLAN tags in-clear configured on the associated MACsec policy must match. That is, if the encapsulation on the interface is 802.1Q or 802.1ad with a single tag, then the value of VLAN tags in-clear in the MACsec policy must be 1. Similarly, if the encapsulation on the interface is 802.1Q or 802.1ad with double tags, then the value of VLAN tags in-clear in the MACsec policy must be 2.

- MACsec support on physical interfaces and subinterfaces is mutually exclusive. To configure MACsec on subinterfaces, clear the MACsec configurations on the corresponding physical interface and conversely.
- The default VLAN tags in-clear value is 1.
- The following MACsec policy parameters must be identical in all subinterfaces in a physical interface:
  - security-policy
  - window-size
  - vlan-tags-in-clear
  - allow-lacp-in-clear
- MACsec on subinterfaces does not support data delay protection.
- We recommend keeping the MACsec session limit on anyline card or fixed port router, including all port-level and subinterface-level MACsec sessions, at 192 for optimal functioning of simultaneous hitless SAK rekey performance.

EAPoL Ether-Type and Destination-Address

In WAN MACsec, when two peers establish an MKA session using the standard EAPoL Ether-Type (0x888E) and destination MAC address (01:80:C2:00:00:03) via the service provider network, the Layer 2 intermediate devices may intercept and consume the EAPoL packets, which in turn can affect the MACsec session establishment between the two endpoints. To overcome this challenge, you can configure an alternate EAPoL Ether-Type, Destination MAC address, or both under the MACsec-enabled interface. For MACsec on subinterfaces, you can configure explicit Ether-Type and Destination MAC address under the subinterfaces; otherwise, the subinterfaces inherit the EAPoL configurations from the parent physical interface.

The alternate EAPoL Ether-Type suported is 0x876F. To configure an alternate EAPoL Ether-Type, refer Configure EAPoL Ether-Type 0x876F, on page 226.

The alternate EAPoL Destination MAC address supported is the multicast address FF:FF:FF:FF:FF or any nearest bridge group address. To configure an alternate EAPoL Destination-Address, refer Configure EAPoL Destination Address , on page 227.

Configuring and Verifying MACSec Encryption

MACSec can be configured on physical ethernet interfaces or VLAN sub-interfaces. The following section describes procedures for configuring and verifying MACSec configuration in any of the described deployment modes.

1. Creating a MACSec Key Chain.
2. Creating a MACSec Policy.
3. Applying MACSec on a Interface.

Creating a MACsec Key Chain

A MACsec keychain is a collection of keys used to authenticate peers needing to exchange encrypted information. While creating a keychain, we define the key(s), key string with password, the cryptographic algorithm, and the key lifetime.

MACsec Keychain Keyword	Description
Key	The MACsec key or the CKN can be up to 64 characters in length. The key must be of an even number of characters. Entering an odd number of characters will exit the MACsec configuration mode.
Key-string	The MACsec key-string or the CAK can be either 32 characters or 64 characters in length (32 for AES-128, 64 for AES-256).
Lifetime	This field specifies the validity period of a key. It includes a start time, and an expiry time. We recommend you to set the value for expiry time as *infinite*.

Guidelines for Configuring MACsec Keychain

MACsec keychain management has the following configuration guidelines:

- To establish MKA session, ensure that the MACsec key (CKN) and key-string (CAK) match at both ends.
- MKA protocol uses the latest active key available in the Keychain. This key has the latest Start Time from the existing set of currently active keys. You can verify the values using the **show key chain** *keychain-name* command.
- Deletion or expiry of current active key brings down the MKA session resulting in traffic hit. We recommend you to configure the keys with infinite lifetime. If fallback is configured, traffic is safeguarded using fallback on expiry or deletion of primary-keychain active key.
- To achieve successful key rollover (CAK-rollover), the new key should be configured such that it is the latest active key, and kicks-in before the current key expires.
- We recommend an overlap of at least one minute for hitless CAK rollover from current key to new key.
- Start time and Expiry time can be configured with future time stamps, which allows bulk configuration for daily CAK rotation without any intervention of management agent.
- From Cisco IOS XR Software Release 7.1.2 and later, the MACsec key IDs (configured through CLI using the **macsec key** command under the key chain configuration mode) are considered to be case insensitive. These key IDs are stored as uppercase letters. For example, a key ID of value 'FF' and of value 'ff' are considered to be the same, and both these key IDs are now stored in uppercase as 'FF'. Whereas, prior to Release 7.1.2, both these values were treated as case sensitive, and hence considered as two separate key IDs. Hence it is recommended to have unique strings as key IDs for a MACsec key chain to avoid flapping of MACsec sessions. However, the support for this case insensitive IDs is applicable only for the configurations done through CLI, and not for configurations done through Netconf protocol.

  Also, it is recommended to do a prior check of the key IDs before upgrading to Release 7.1.2 or later.

Consider a scenario where two MACsec key IDs with the same set of characters (say, ff and FF) are configured under the same key chain.

```
key chain 1
 macsec
  key ff
   lifetime 02:01:01 may 18 2020 infinite
  !
  key FF
   lifetime 01:01:01 may 18 2020 infinite
```

When you upgrade to Release 7.1.2 or later, only one of these key IDs is retained. That is 'FF', the one that was applied second in this example.

SUMMARY STEPS

1. Enter the global configuration mode and provide a name for the MACsec keychain; for example, mac_chain.
2. Enter the MACsec mode.
3. Provide a name for the MACsec key.
4. Enter the key string and the cryptographic algorithm to be used for the key.
5. Enter the validity period for the MACsec key (CKN) also known as the lifetime period.
6. Commit your configuration.

DETAILED STEPS

Procedure

Step 1 Enter the global configuration mode and provide a name for the MACsec keychain; for example, mac_chain.

Example:

```
RP/0/RSP0/CPU0:router(config)#key chain mac_chain
```

Step 2 Enter the MACsec mode.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_chain)#macsec
```

Step 3 Provide a name for the MACsec key.

The key can be up to 64 characters in length. The key must be of an even number of characters. Entering an odd number of characters will exit the MACsec configuration mode.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_chain-MacSec)#key 1234abcd5678
```

You can also configure a fall-back pre-shared key(PSK) to ensure that a PSK is always available to perform MACsec encryption and decryption. The fallback PSK along with the primary PSK ensures that the session remains active even if the primary PSK is mismatched or there is no active key for the primary PSK.

The configured key is the CKN that is exchanged between the peers.

See the guidelines section to know more about the need for a unique key ID for a MACsec key chain.



Note If you are configuring MACsec to inter-operate with a MACsec server that is running software prior to Cisco IOS XR Release 6.1.3, then ensure that the MACsec key length is of 64 characters. You can add extra zero characters to the MACsec key so that the length of 64-characters is achieved. If the key length is lesser than 64 characters, authentication will fail.

Step 4 Enter the key string and the cryptographic algorithm to be used for the key.

Example:

The key string is the CAK that is used for ICV validation by the MKA protocol.

```
! For AES 128-bit encryption

RP/0/RSP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#key-string 12345678123456781234567812345678
 cryptographic-algorithm AES-128-CMAC

! For AES 256-bit encryption

RP/0/RSP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#key-string
12345678123456781234567812345678123456781234567812345678123456781234567812345678 cryptographic
-algorithm AES-256-CMAC
```



Note In this example, we have used the AES 256-bit encryption algorithm, and therefore, the key string is 64 hexadecimal characters in length. A 256-bit encryption algorithm uses a larger key that requires more rounds of hacking to be cracked. 256-bit algorithms provide better security against large mass security attacks, and include the security provided by 128-bit algorithms.

Step 5 Enter the validity period for the MACsec key (CKN) also known as the lifetime period.

The lifetime period can be configured, with a duration in seconds, as a validity period between two dates (for example, Jan 01 2014 to Dec 31 2014), or with infinite validity.

The key is valid from the time you configure (in HH:MM:SS format). Duration is configured in seconds.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#lifetime 05:00:00 01
January 2015 duration 1800
```

An example of configuring the lifetime for a defined period:

```
RP/0/RSP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#lifetime 05:00:00 20
```

```
february 2015 12:00:00 30 september 2015
```

An example of configuring the lifetime as infinite:

```
RP/0/RSP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#lifetime
05:00:00 01 January 2015 infinite
```



Note When a key has expired, the MACsec session is torn down and running the **show macsec mka session** command does not display any information. If you run the **show macsec mka interface detail** command, the output displays ***** No Active Keys Present ***** in the PSK information.

Step 6 Commit your configuration.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_chain-MacSec-1234abcd5678)#commit
```

This completes the configuration of the MACsec keychain.

Creating a User-Defined MACsec Policy

SUMMARY STEPS

1. Enter the global configuration mode, and enter a name (mac_policy) for the MACsec policy.
2. Configure the cipher suite to be used for MACsec encryption.
3. Configure the confidentiality offset for MACsec encryption.
4. Enter the key server priority.
5. Configure the security policy parameters, either Must-Secure or Should-Secure.
6. Configure the replay protection window size.
7. Configure the ICV for the frame arriving on the port.
8. Commit your configuration and exit the global configuration mode.
9. Confirm the MACsec policy configuration.

DETAILED STEPS

Procedure

Step 1 Enter the global configuration mode, and enter a name (mac_policy) for the MACsec policy.

Example:

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# macsec-policy mac_policy
```

Step 2 Configure the cipher suite to be used for MACsec encryption.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_policy)# cipher-suite GCM-AES-XPN-256
RP/0/RSP0/CPU0:router(config-mac_policy)#GCM-AES-128
GCM-AES-256
GCM-AES-XPN-128
GCM-AES-XPN-256
```

In this example, the GCM-AES-XPN-256 encryption algorithm is used. A 256-bit encryption algorithm uses a larger key that requires more rounds of hacking to be cracked. 256-bit algorithms provide better security against large mass security attacks, and include the security provided by 128-bit algorithms. Extended Packet Numbering (XPN) is used to reduce the number of key rollovers while data is sent over high speed links. It is therefore highly recommended to use GCM-AES-XPN-256 encryption algorithm for higher data ports.



Note Use of non-XPN ciphers on high-speed links (40G and above) is not recommended, as frequent packet number exhaustion and rekey operations may lead to transient traffic loss.

Step 3 Configure the confidentiality offset for MACsec encryption.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_policy)# conf-offset CONF-OFFSET-30
```



Note We recommend to change the offset value of the **conf-offset** <*offset_value*> command (MACsec encryption command) in the router only when the port is in **admin down** state (that is, when the interface is shut down). Changing the offset value otherwise may result in traffic loss.

Step 4 Enter the key server priority.

You can enter a value between 0-255. Lower the value, higher the preference to be selected as the key server.

In this example, a value of 0 configures the router as the key server, while the other router functions as a key client. The key server generates and maintains the SAK between the two routers. The default key server priority value is 16.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_policy)# key-server-priority 0
```

Step 5 Configure the security policy parameters, either Must-Secure or Should-Secure.

Must-Secure: Must-Secure imposes only MACsec encrypted traffic to flow. Hence, until MKA session is not secured, traffic will be dropped.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_policy)# security-policy must-secure
```

Should-Secure: Should-Secure allows unencrypted traffic to flow until MKA session is secured. After the MKA session is secured, Should-Secure policy imposes only encrypted traffic to flow.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_policy)# security-policy should-secure
```

Table 34: MACsec Security Policies

MKA		Secured MKA Session	Unsecured MKA Session
Security Policy	Must-secure	Encrypted traffic	Traffic drop (no Tx and no Rx)
	Should-secure	Encrypted traffic	Plain text or unencrypted traffic

Step 6 Configure the replay protection window size.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_policy)# window-size 64
```

This dictates the maximum out-of-sequence frames that are accepted. You can configure a value between 0 and 1024.

Step 7 Configure the ICV for the frame arriving on the port.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_policy)# include-icv-indicator
```

This parameter configures inclusion of the optional ICV Indicator as part of the transmitted MACsec Key Agreement PDU (MKPDU). This configuration is necessary for MACsec to interoperate with routers that run software prior to IOS XR version 6.1.3. This configuration is also important in a service provider WAN setup where MACsec interoperates with other vendor MACsec implementations that expect ICV indicator to be present in the MKPDU.

Step 8 Commit your configuration and exit the global configuration mode.

Example:

```
RP/0/RSP0/CPU0:router(config-mac_policy)# exit
RP/0/RSP0/CPU0:router(config)# commit
RP/0/RSP0/CPU0:router(config)# exit
```

Step 9 Confirm the MACsec policy configuration.

Example:

```
RP/0/RSP0/CPU0:router# show running-config macsec-policy

macsec-policy mac_policy
conf-offset CONF-OFFSET-30
security-policy must-secure
window-size 64
cipher-suite GCM-AES-XPN-256
key-server-priority 0
include-icv-indicator
```

This completes the configuration of the MACsec policy.

Note

- Small packets might be dropped when Data Delay Protection (DDP) is enabled on many MACsec enabled interfaces of a scaled setup. To avoid this, enable DDP only on the interfaces which are absolutely necessary.
- For Cisco ASR 9000 Series Routers to interoperate with Cisco ASR9000 Series Routers that are older than Release 6.2.3, configure a user defined MACsec policy with the `policy-exception lacp-in-clear` command to bring up the MKA sessions over bundle interfaces running in LACP modes.

MACsec SAK Rekey Interval

From Cisco IOS XR Software Release 6.3.3 and later, you can set a timer value to rekey the MACsec secure association key (SAK) at a specified interval. This periodic refresh of SAK ensures that data encryption key is frequently updated. The configuration is effective on the node acting as a key server.

To set the rekey interval, use the **sak-rekey-interval** command in macsec-policy configuration mode. The timer ranges from 60 to 2,592,000 seconds, the default being OFF.

Configuration Example

```
Router#configure
Router(config)#macsec-policy test-policy
Router(config-macsec-policy)#sak-rekey-interval 120
Router(config-macsec-policy)#commit
```

Running Configuration

```
macsec-policy test-policy
 sak-rekey-interval 120
 !
```

Associated Command

sak-rekey-interval

MACsec Policy Exceptions

By default, the MACsec security policy uses **must-secure** option, that mandates data encryption. Hence, the packets cannot be sent in clear-text format. To optionally bypass the MACsec encryption or decryption for Link Aggregation Control Protocol (LACP) packets, and to send the packets in clear-text format, use the **policy-exception lacp-in-clear** command in macsec-policy configuration mode. This functionality is beneficial in scenarios such as, in a network topology with three nodes, where bundles are terminated at the middle node, whereas MACsec is terminated at the end nodes.

This MACsec policy exception is also beneficial in interoperability scenarios where the node at the other end expects the data packets to be in clear text.

From Cisco IOS XR Software Release 7.3.1 and later, an alternative option, **allow**, is introduced under the macsec-policy configuration mode, that allows packets to be sent in clear-text format. You can use the **allow lacp-in-clear** command for LACP packets.

How to Create MACsec Policy Exception

Note The **policy-exception lacp-in-clear** command under macsec-policy configuration mode is deprecated. Hence, it is recommended to use the **allow lacp-in-clear** command instead, to allow LACP packets in clear-text format.

Configuration Example

Using the **policy-exception** command:

```
Router#configure
Router(config)#macsec-policy P1
Router(config-macsec-policy-P1)#policy-exception lacp-in-clear
Router(config-macsec-policy-P1)#commit
```

Using the **allow** command:

```
Router#configure
Router(config)#macsec-policy P1
Router(config-macsec-policy-P1)#allow lacp-in-clear
Router(config-macsec-policy-P1)#commit
```

Running Configuration

With the **policy-exception** command:

```
Router#show run macsec-policy P1
macsec-policy P1
  policy-exception lacp-in-clear
  security-policy should-secure
  include-icv-indicator
  sak-rekey-interval seconds 120
!
```

With the **allow** command:

```
Router#show run macsec-policy P1
macsec-policy P1
  allow lacp-in-clear
  security-policy should-secure
  include-icv-indicator
  sak-rekey-interval seconds 120
!
```

Associated Commands

- **policy-exception lacp-in-clear**
- **allow lacp-in-clear**

Applying MACsec Configuration on an Interface

Guidelines for MACsec Interface Configuration

- Configure different keychains for primary and fallback PSKs.
- We do not recommend to update both primary and fallback PSKs simultaneously, because fallback PSK is intended to recover MACsec session on primary key mismatch.
- When using MACsec, we recommend you adjust the maximum transmission unit (MTU) of an interface to accommodate the MACsec overhead. Configuring MTU value on an interface allows protocols to do MTU negotiation including MACsec overhead. For instance, if the default MTU is 1514 bytes, configure the MTU to 1546 bytes (1514 + 32).
- The minimum MTU for IS-IS protocol on the MACsec interface is 1546 bytes.
- To enable MACsec on bundles:
  - Enable MACsec on all bundle members.
  - MACsec peers running IOS-XR version 24.1.1 or higher:
    - For routing protocols running on the bundle interface, configure **impose-overhead-on-bundle** in the MACsec policy to adjust the bundle interface MTU with MACsec overhead.
  - MACsec peers running IOS-XR versions prior to 24.1.1:
    - We recommend configuring the maximum possible MTU on the bundle interface.
    - The MTU configurations must account for the maximum packet size of the protocols running on the bundle interface and 32 bytes of MACsec overhead.
    - For IS-IS protocol running on the bundle interface, hello-padding must be disabled.

Tip You can programmatically view the MACsec configuration using the `openconfig-macsec.yang` OpenConfig data model. To get started with using data models, see *Programmability Configuration Guide for Cisco ASR 9000 Series Routers*.

MACsec PSK Configuration on an Interface

```
Router#configure terminal
Router(config)#interface Te0/3/0/1/4
Router(config-if)#macsec psk-keychain kc policy mac_policy
```

To apply MACsec configuration on a physical interface without the MACsec policy, use the following command:

```
Router(config-if)#macsec psk-keychain kc
```

MACsec Fallback PSK Configuration on an Interface

It is optional to configure a fallback PSK. If a fallback PSK is configured, the fallback PSK along with the primary PSK ensures that the session remains active even if the primary PSK is mismatched, or there is no active key for the primary PSK.

```
Router(config-if)#macsec psk-keychain kc fallback-psk-keychain fallback_kc policy mac_policy
Router(config-if)#commit
```

Configuring and Verifying MACsec Encryption on Physical Interfaces

Enabling MACsec encryption on physical interfaces involves the following steps:

Configuration

1. Creating a MACsec Key Chain.
2. Creating a MACsec Policy.
3. Applying MACsec on a interface:

```
Router# configure
Router(config)# interface HundredGigE 0/5/0/16
Router(config-subif)# ipv4 address 192.168.16.1 255.255.255.0
Router(config-subif)# macsec psk-keychain kc fallback-psk-keychain fb
Router(config-subif)# commit
```

Running Configuration

Sub-Interface Configurations:

```
Router# show running-config interface HundredGigE 0/5/0/16
interface HundredGigE0/5/0/16
 ipv4 address 192.168.16.1 255.255.255.0
 macsec psk-keychain kc fallback-psk-keychain fb
 !
```

Verification

```
Router# show macsec mka summary
NODE: node0_5_CPU0
================================================================================================
   Interface-Name     Status     Cipher-Suite       KeyChain       PSK/EAP      CKN
================================================================================================
   Hu0/5/0/16     Secured  GCM-AES-XPN-256         kc          PRIMARY     1234
   Hu0/5/0/30     Secured  GCM-AES-XPN-256         kc          PRIMARY     1234
```

```
Router# show macsec mka interface detail
Interface Name : HundredGigE0/5/0/16.100
    Interface Namestring     : HundredGigE0/5/0/16.100
    Interface short name     : Hu0/5/0/16.100
    Interface handle         : 0x2800b00
    Interface number         : 0x2800b00
    MacSecControlledIfh      : 0x2800b08
    MacSecUnControlledIfh    : 0x2800b10
    Interface MAC            : e069.bafd.e3a0
    Ethertype                : 888E
    EAPoL Destination Addr   : 0180.c200.0003
    MACsec Shutdown          : FALSE
    Config Received          : TRUE
    IM notify Complete       : TRUE
    MACsec Power Status      : Allocated
    Interface CAPS Add       : TRUE
    RxSA CAPS Add            : TRUE
    TxSA CAPS Add            : TRUE
    IM notify with VLAN Info : TRUE
    Supported VLAN encaps    : TRUE
    SecTAG Offset validation : TRUE
```

```
VLAN                      : Outer tag (etype=0x8100, id=100, priority=0, cfi=0)
Principal Actor           : Primary
MKA PSK Info
  Key Chain Name          : kc
  MKA Cipher Suite        : AES-256-CMAC
  CKN                     : 12 34
MKA fallback_PSK Info
  fallback keychain Name  : - NA -
Policy                    : mp-SF1
SKS Profile               : N/A
Traffic Status            : Protected
Rx SC 1
  Rx SCI                  : e069bafde3a80064
  Rx SSCI                 : 1
  Peer MAC                : e0:69:ba:fd:e3:a8
  Is XPN                  : YES
  SC State                : Provisioned
  SAK State[0]            : Provisioned
  Rx SA Program Req[0]    : 2023 Oct 27 05:41:51.701
  Rx SA Program Rsp[0]    : 2023 Oct 27 05:41:51.705
  SAK Data
    SAK[0]                : ***
    SAK Len               : 32
    SAK Version           : 1
    HashKey[0]            : ***
    HashKey Len           : 16
    Conf offset           : 0
    Cipher Suite          : GCM-AES-XPN-256
    CtxSalt[0]            : c2 b0 88 9d d6 c0 9d 3f 0a b7 99 37
    CtxSalt Len           : 12
    ssci                  : 1

Tx SC
  Tx SCI                  : e069bafde3a00064
  Tx SSCI                 : 2
  Active AN               : 0
  Old AN                  : 255
  Is XPN                  : YES
  Next PN                 : 1, 0, 0, 0
  SC State                : Provisioned
  SAK State[0]            : Provisioned
  Tx SA Program Req[0]    : 2023 Oct 27 05:41:51.713
  Tx SA Program Rsp[0]    : 2023 Oct 27 05:41:51.715
  SAK Data
    SAK[0]                : ***
    SAK Len               : 32
    SAK Version           : 1
    HashKey[0]            : ***
    HashKey Len           : 16
    Conf offset           : 0
    Cipher Suite          : GCM-AES-XPN-256
    CtxSalt[0]            : c2 b0 88 9e d6 c0 9d 3f 0a b7 99 37
    CtxSalt Len           : 12
    ssci                  : 2
```

For detailed information on verifying MACsec encryption, refer Verifying MACsec Encryption on IOS XR, on page 229.

Configuring and Verifying MACsec Encryption on VLAN Subinterfaces

Enabling MACsec encryption on subinterfaces involves the following steps:

1. Creating a MACsec Key Chain.

2. Creating a MACsec Policy.
3. Applying MACsec on a Subinterface.

MACsec on VLAN Subinterfaces with Single Tag

Configuration

1. Creating a MACsec Key Chain:

```
Router# configure
Router(config)# key chain kc
Router(config-kc)# macsec
Router(config-kc-macsec)# key 1234
Router(config-kc-macsec-1234)# key-string
12345678123456781234567812345678123456781234567812345678 cryptographic-algorithm
 aes-256-cmac
Router(config-kc-macsec-1234)# lifetime 05:00:00 1 January 2023 infinite
Router(config-kc-macsec-1234)# commit
```

2. Creating a MACsec Policy:

```
Router# configure
Router(config)# macsec-policy mp-SF1
RRouter(config-macsec-policy)# vlan-tags-in-clear 1
/* The VLAN tagging in the MACsec policy must match the encapsulation on the interface
*/
Router(config-macsec-policy)# commit
```

3. Applying MACsec on a Subinterface:

```
Router# configure
Router(config)# interface HundredGigE 0/5/0/16.100
Router(config-subif)# encapsulation dot1q 100
Router(config-subif)# ipv4 address 192.168.16.1 255.255.255.0
Router(config-subif)# macsec psk-keychain kc policy mp-SF1
Router(config-subif)# commit
```

Running Configuration

MACsec Key Chain:

```
Router# show running-config psk-keychain kc
key chain kc
 macsec
  key 1234
   key-string password
11584B5643475D5B5C7B79777C6663754B56445055030F0F0B055C504C430F0F0F020006005E0D515F0905574753520C53575D72181B5F4E5D46405858517C7C7C
 cryptographic-algorithm aes-256-cmac
   lifetime 05:00:00 january 01 2023 infinite
  !
 !
!
```

MACsec Policy:

```
Router# show running-config macsec-policy mp-SF1
macsec-policy mp-SF1
 ...
```

```
 vlan-tags-in-clear 1
!
```

Sub-Interface Configurations:

```
Router# show running-config interface HundredGigE 0/5/0/16.100
interface HundredGigE0/5/0/16.100
 ipv4 address 192.168.16.1 255.255.255.0
 macsec psk-keychain kc policy mp-SF1
 encapsulation dot1q 100
!
```

Verification

```
Router# show macsec mka summary
NODE: node0_5_CPU0
================================================================================================
   Interface-Name     Status     Cipher-Suite       KeyChain       PSK/EAP      CKN
================================================================================================
   Hu0/5/0/16.100     Secured  GCM-AES-XPN-256         kc          PRIMARY     1234
   Hu0/5/0/30.200     Secured  GCM-AES-XPN-256         kc          PRIMARY     1234
```

```
Router# show macsec policy mp-SF1 detail
Policy Name                 : mp-SF1
      Cipher Suite          : GCM-AES-XPN-256
      Key-Server Priority   : 10
      Window Size           : 64
      Conf Offset           : 0
      Replay Protection     : TRUE
      Delay Protection      : FALSE
      Security Policy       : Must Secure
      Vlan Tags In Clear    : 1
      LACP In Clear         : FALSE
      Pause Frame In Clear  : FALSE
      Sak Rekey Interval    : OFF
      Include ICV Indicator : FALSE
      Use Eapol PAE in ICV  : FALSE
      Disable Suspend On Request    : FALSE
      Disable Suspend For           : FALSE
      Enable legacy fallback        : FALSE
      SKS Profile                   : N/A
      Max AN                        : 3
      Impose Overhead on Bundle     : FALSE
```

```
Router# show macsec mka interface detail
Interface Name : HundredGigE0/5/0/16.100
    Interface Namestring     : HundredGigE0/5/0/16.100
    Interface short name     : Hu0/5/0/16.100
    Interface handle         : 0x2800b00
    Interface number         : 0x2800b00
    MacSecControlledIfh      : 0x2800b08
    MacSecUnControlledIfh    : 0x2800b10
    Interface MAC            : e069.bafd.e3a0
    Ethertype                : 888E
    EAPoL Destination Addr   : 0180.c200.0003
    MACsec Shutdown          : FALSE
    Config Received          : TRUE
    IM notify Complete       : TRUE
    MACsec Power Status      : Allocated
    Interface CAPS Add       : TRUE
    RxSA CAPS Add            : TRUE
    TxSA CAPS Add            : TRUE
```

```
IM notify with VLAN Info : TRUE
Supported VLAN encaps    : TRUE
SecTAG Offset validation : TRUE
VLAN                     : Outer tag (etype=0x8100, id=100, priority=0, cfi=0)
Principal Actor          : Primary
MKA PSK Info
  Key Chain Name         : kc
  MKA Cipher Suite       : AES-256-CMAC
  CKN                    : 12 34
MKA fallback_PSK Info
  fallback keychain Name : - NA -
Policy                   : mp-SF1
SKS Profile              : N/A
Traffic Status           : Protected
Rx SC 1
  Rx SCI                 : e069bafde3a80064
  Rx SSCI                : 1
  Peer MAC               : e0:69:ba:fd:e3:a8
  Is XPN                 : YES
  SC State               : Provisioned
  SAK State[0]           : Provisioned
  Rx SA Program Req[0]   : 2023 Oct 27 05:41:51.701
  Rx SA Program Rsp[0]   : 2023 Oct 27 05:41:51.705
  SAK Data
    SAK[0]               : ***
    SAK Len              : 32
    SAK Version          : 1
    HashKey[0]           : ***
    HashKey Len          : 16
    Conf offset          : 0
    Cipher Suite         : GCM-AES-XPN-256
    CtxSalt[0]           : c2 b0 88 9d d6 c0 9d 3f 0a b7 99 37
    CtxSalt Len          : 12
    ssci                 : 1

Tx SC
  Tx SCI                 : e069bafde3a00064
  Tx SSCI                : 2
  Active AN              : 0
  Old AN                 : 255
  Is XPN                 : YES
  Next PN                : 1, 0, 0, 0
  SC State               : Provisioned
  SAK State[0]           : Provisioned
  Tx SA Program Req[0]   : 2023 Oct 27 05:41:51.713
  Tx SA Program Rsp[0]   : 2023 Oct 27 05:41:51.715
  SAK Data
    SAK[0]               : ***
    SAK Len              : 32
    SAK Version          : 1
    HashKey[0]           : ***
    HashKey Len          : 16
    Conf offset          : 0
    Cipher Suite         : GCM-AES-XPN-256
    CtxSalt[0]           : c2 b0 88 9e d6 c0 9d 3f 0a b7 99 37
    CtxSalt Len          : 12
    ssci                 : 2
```

For detailed information on verifying MACsec encryption, refer Verifying MACsec Encryption on IOS XR, on page 229.

MACsec on VLAN Subinterfaces with Double Tag

Configuration

1. Creating a MACsec Key Chain:

```
Router# configure
Router(config)# key chain kc
Router(config-kc)# macsec
Router(config-kc-macsec)# key 1234
Router(config-kc-macsec-1234)# key-string
12345678123456781234567812345678123456781234567812345678 cryptographic-algorithm
 aes-256-cmac
Router(config-kc-macsec-1234)# lifetime 05:00:00 1 January 2023 infinite
Router(config-kc-macsec-1234)# commit
```

2. Creating a MACsec Policy:

```
Router# configure
Router(config)# macsec-policy mp-SF1
RRouter(config-macsec-policy)# vlan-tags-in-clear 2
/* The VLAN tagging in the MACsec policy must match the encapsulation on the interface
*/
Router(config-macsec-policy)# commit
```

3. Applying MACsec on a Subinterface:

```
Router# configure
Router(config)# interface HundredGigE 0/5/0/30.200
Router(config-subif)# encapsulation dot1ad 200 dot1q 300
Router(config-subif)# ipv4 address 192.168.30.1 255.255.255.0
Router(config-subif)# macsec psk-keychain kc policy mp-SF2
Router(config-subif)# commit
```

Running Configuration

MACsec Key Chain:

```
Router# show running-config psk-keychain kc
key chain kc
 macsec
  key 1234
   key-string password
11584B5643475D5B5C7B79777C6663754B56445055030F0F0B055C504C430F0F0F020006005E0D515F0905574753520C53575D72181B5F4E5D46405858517C7C7C
 cryptographic-algorithm aes-256-cmac
   lifetime 05:00:00 january 01 2023 infinite
  !
 !
!
```

MACsec Policy:

```
Router# show running-config macsec-policy mp-SF2
macsec-policy mp-SF2
 ...
 vlan-tags-in-clear 2!
```

Subinterface Configurations:

```
Router# show running-config interface HundredGigE 0/5/0/30.200
interface HundredGigE0/5/0/30.200
```

```
 ipv4 address 192.168.30.1 255.255.255.0
 macsec psk-keychain kc policy mp-SF2
 encapsulation dot1ad 200 dot1q 300
```

Verification

```
Router# show macsec mka summary
NODE: node0_5_CPU0
================================================================================================
   Interface-Name     Status     Cipher-Suite       KeyChain       PSK/EAP      CKN
================================================================================================
   Hu0/5/0/16.100     Secured  GCM-AES-XPN-256         kc           PRIMARY     1234
   Hu0/5/0/30.200     Secured  GCM-AES-XPN-256         kc           PRIMARY     1234


Router# show macsec policy mp-SF2 detail
Policy Name                 : mp-SF2
      Cipher Suite          : GCM-AES-XPN-256
      Key-Server Priority   : 20
      Window Size           : 64
      Conf Offset           : 0
      Replay Protection     : TRUE
      Delay Protection      : FALSE
      Security Policy       : Must Secure
      Vlan Tags In Clear    : 2
      LACP In Clear         : FALSE
      Pause Frame In Clear  : FALSE
      Sak Rekey Interval    : OFF
      Include ICV Indicator : FALSE
      Use Eapol PAE in ICV  : FALSE
      Disable Suspend On Request    : FALSE
      Disable Suspend For           : FALSE
      Enable legacy fallback        : FALSE
      SKS Profile                   : N/A
      Max AN                        : 3
      Impose Overhead on Bundle     : FALSE


Router# show macsec mka interface detail
Interface Name : HundredGigE0/5/0/30.200
    Interface Namestring     : HundredGigE0/5/0/30.200
    Interface short name     : Hu0/5/0/30.200
    Interface handle         : 0x2800b30
    Interface number         : 0x2800b30
    MacSecControlledIfh      : 0x2800b38
    MacSecUnControlledIfh    : 0x2800b40
    Interface MAC            : e069.bafd.e410
    Ethertype                : 888E
    EAPoL Destination Addr   : 0180.c200.0003
    MACsec Shutdown          : FALSE
    Config Received          : TRUE
    IM notify Complete       : TRUE
    MACsec Power Status      : Allocated
    Interface CAPS Add       : TRUE
    RxSA CAPS Add            : TRUE
    TxSA CAPS Add            : TRUE
    IM notify with VLAN Info : TRUE
    Supported VLAN encaps    : TRUE
    SecTAG Offset validation : TRUE
    VLAN                     : Outer tag (etype=0x88a8, id=200, priority=0, cfi=0)
                             : Inner tag (etype=0x8100, id=300, priority=0, cfi=0)
    Principal Actor          : Primary
    MKA PSK Info
      Key Chain Name         : kc
```

```
  MKA Cipher Suite       : AES-256-CMAC
  CKN                    : 12 34
MKA fallback_PSK Info
  fallback keychain Name : - NA -
Policy                   : mp-SF2
SKS Profile              : N/A
Traffic Status           : Protected
Rx SC 1
  Rx SCI                 : e069bafde41800c8
  Rx SSCI                : 1
  Peer MAC               : e0:69:ba:fd:e4:18
  Is XPN                 : YES
  SC State               : Provisioned
  SAK State[0]           : Provisioned
  Rx SA Program Req[0]   : 2023 Oct 27 05:44:01.270
  Rx SA Program Rsp[0]   : 2023 Oct 27 05:44:01.274
  SAK Data
    SAK[0]               : ***
    SAK Len              : 32
    SAK Version          : 1
    HashKey[0]           : ***
    HashKey Len          : 16
    Conf offset          : 0
    Cipher Suite         : GCM-AES-XPN-256
    CtxSalt[0]           : 02 52 27 e4 ba 7f 16 62 52 d8 a6 e8
    CtxSalt Len          : 12
    ssci                 : 1

Tx SC
  Tx SCI                 : e069bafde41000c8
  Tx SSCI                : 2
  Active AN              : 0
  Old AN                 : 255
  Is XPN                 : YES
  Next PN                : 1, 0, 0, 0
  SC State               : Provisioned
  SAK State[0]           : Provisioned
  Tx SA Program Req[0]   : 2023 Oct 27 05:44:01.282
  Tx SA Program Rsp[0]   : 2023 Oct 27 05:44:01.284
  SAK Data
    SAK[0]               : ***
    SAK Len              : 32
    SAK Version          : 1
    HashKey[0]           : ***
    HashKey Len          : 16
    Conf offset          : 0
    Cipher Suite         : GCM-AES-XPN-256
    CtxSalt[0]           : 02 52 27 e7 ba 7f 16 62 52 d8 a6 e8
    CtxSalt Len          : 12
    ssci                 : 2
```

For detailed information on verifying MACsec encryption, refer Verifying MACsec Encryption on IOS XR, on page 229.

Configure EAPoL Ether-Type 0x876F

Enabling EAPoL Ether-Type 0x876F involves the following steps:

Configuration

1. Creating a MACsec Key Chain.
2. (Optional) Creating a MACsec Policy.

3. Configure EAPoL ether-type.

```
Router(config)# interface HundredGigE0/1/0/2
Router(config-if)# eapol eth-type 876F
Router(config-if)# commit
```

4. Applying MACsec on a interface.

```
Router(config)# interface HundredGigE0/1/0/2
Router(config-if)# macsec psk-keychain kc fallback-psk-keychain fb
Router(config-if)# commit
```

Running Configuration

```
Router# show running-config interface HundredGigE0/1/0/2
interface HundredGigE0/1/0/2
  eapol eth-type 876F
  macsec psk-keychain kc fallback-psk-keychain fb
!
```

Verification

```
Router# show macsec mka interface  HundredGigE0/1/0/2 detail | i Ethertype
Ethertype                : 876F

Router# show macsec mka session interface HundredGigE0/1/0/2.1
===============================================================================================

   Interface-Name        Local-TxSCI       #Peers   Status   Key-Server   PSK/EAP      CKN

===============================================================================================

     Hu0/1/0/2        0201.9ab0.77cd/0001     1      Secured      YES      PRIMARY     1234

     Hu0/1/0/2        0201.9ab0.77cd/0001     1      Active       YES     FALLBACK     9999
```

Configure EAPoL Destination Address

Configuring EAPoL destination address involves the following steps:

Broadcast Address

The EAPoL destination address is set to broadcast address, FF:FF:FF:FF:FF to ensure the underlying L2 network will flood the EAPoL packets to all receivers.

Configuration

1. Creating a MACsec Key Chain.
2. (Optional) Creating a MACsec Policy.
3. Configure EAPoL destination address.

```
Router(config)# interface HundredGigE0/1/0/2
Router(config-if)# eapol destination-address broadcast-address
Router(config-if)# commit
```

4. Applying MACsec on a interface.

```
Router(config)# interface HundredGigE0/1/0/2
Router(config-if)# macsec psk-keychain kc fallback-psk-keychain fb
Router(config-if)# commit
```

Running Configuration

```
Router# show running-config interface HundredGigE0/1/0/2
 eapol destination-address ffff.ffff.ffff
 macsec psk-keychain kc fallback-psk-keychain fb
!
```

Verification

```
Router# show macsec mka interface  HundredGigE0/1/0/2 detail  | i EAPoL
    EAPoL Destination Addr   : ffff.ffff.ffff
```

```
Router# show macsec mka session interface HundredGigE0/1/0/2
================================================================================================

   Interface-Name        Local-TxSCI       #Peers   Status   Key-Server   PSK/EAP      CKN

================================================================================================

     Hu0/1/0/2       02df.3638.d568/0001     1      Secured      YES      PRIMARY     1234

     Hu0/1/0/2       02df.3638.d568/0001     1      Active       YES     FALLBACK     9999
```

EAPoL Bridge Group Address

The EAPoL destination address can be set to the nearest bridge group address, for example 01:80:C2:00:00:00.

The following example shows EAPoL destination address configuration on a physical interface, which is inherited by the MACsec enabled subinterface.

Configuration

1. Creating a MACsec Key Chain.
2. (Optional) Creating a MACsec Policy.
3. Configure EAPoL destination address to a MACsec enabled physical interface.

```
Router(config)# interface HundredGigE0/1/0/1
Router(config-if)# eapol destination-address bridge-group-address 0180.c200.0000
Router(config-if)# commit
```

4. Configure MACsec on a subinterface.

```
Router(config)# interface HundredGigE0/1/0/1.1
Router(config-subif)# encapsulation dot1q 1
Router(config-subif)# macsec psk-keychain kc fallback-psk-keychain fb
outer(config-subif)# commit
```

Running Configuration

```
Router# show running-config interface Hu0/1/0/1
interface HundredGigE0/1/0/1
eapol destination-address 0180.c200.0000
```

```
Router# show running-config interface HundredGigE0/1/0/1.1
interface HundredGigE0/1/0/0.1
  macsec psk-keychain kc fallback-psk-keychain fb
  encapsulation dot1q 1
!
```

Verification

```
Router# show macsec mka interface  HundredGigE0/1/0/1.1 detail  | i EAPoL
    EAPoL Destination Addr   : 0180.c200.0000

Router# show macsec mka session interface HundredGigE0/1/0/1.1
==============================================================================================

   Interface-Name        Local-TxSCI       #Peers   Status   Key-Server   PSK/EAP      CKN

==============================================================================================

     Hu0/1/0/1.1       0201.9ab0.85af/0001     1      Secured      YES       PRIMARY
1234
     Hu0/1/0/1.1       0201.9ab0.85af/0001     1      Active       YES      FALLBACK
9999
```

Verifying MACsec Encryption on IOS XR

MACsec encryption on IOS XR can be verified by running relevant commands in the Privileged Executive Mode. The verification steps are the same for MACsec encryption on L2VPN or L3VPN network.

Note With the introduction of active fallback functionality in Cisco IOS XR Software Release 7.1.2 (Release 6.7.2 for 32-bit Cisco IOS XR platforms), the output of various MACsec show commands include the fallback PSK entry as well.

To verify if MACsec encryption has been correctly configured, follow these steps.

SUMMARY STEPS

1. Verify the MACsec policy configuration.
2. Verify the MACsec configuration on the respective interface.
3. Verify whether the interface of the router is peering with its neighbor after MACsec configuration
4. Verify whether the MKA session is secured with MACsec on the respective interface.
5. Verify the MACsec session counter statistics.

DETAILED STEPS

Procedure

Step 1 Verify the MACsec policy configuration.

Example:

```
RP/0/RSP0/CPU0:router#show macsec policy mac_policy

==========================================================================

Policy      Cipher         Key-Svr     Window  Conf

name        Suite          Priority    Size    Offset
```

```
================================================================================

mac_policy GCM-AES-XPN-256  0          64      30
```

If the values you see are different from the ones you configured, then check your configuration by running the **show run macsec-policy** command.

Step 2 Verify the MACsec configuration on the respective interface.

You can verify the MACsec encryption on the configured interface bundle (MPLS network), P2MP interface (VPLS network), or VLAN sub-interface (EoMPLS PW network).

Example:

Before the introduction of active fallback functionality:

```
RP/0/RSP0/CPU0:router#show macsec mka summary

NODE: node0_0_CPU0

================================================================
Interface     Status    Cipher Suite      KeyChain
================================================================

Fo0/0/0/1/0   Secured   GCM-AES-XPN-256   mac_chain

Total MACSec Sessions : 1
     Secured Sessions : 1
     Pending Sessions : 0

RP/0/RSP0/CPU0:router#show macsec mka session interface Fo0/0/0/1/0
=============================================================================
   Interface-Name        Local-TxSCI       #Peers   Status   Key-Server
=============================================================================
   Fo0/0/0/1/0        d46d.5023.3709/0001     1     Secured      YES


! If sub-interfaces are configured, the output would be as follows:

RP/0/RSP0/CPU0:router#show macsec mka session interface Fo0/0/0/1/1.8
=====================================================================
      Interface         Local-TxSCI       # Peers   Status  Key-Server
=====================================================================
  Fo0/0/0/1/1.8      e0ac.f172.4124/001d      1     Secured     Yes
```

With the introduction of active fallback functionality:

The following is a sample output that displays active fallback PSK entry as well:

```
RP/0/RSP0/CPU0:router#show macsec mka summary

NODE: node0_0_CPU0
=============================================================================================
   Interface-Name     Status     Cipher-Suite       KeyChain       PSK/EAP      CKN
=============================================================================================
     Fo0/0/0/1/0      Secured  GCM-AES-XPN-256      mac_chain      PRIMARY      5555
     Fo0/0/0/1/0      Active   GCM-AES-XPN-256      mac_chain_fb   FALLBACK     5556

Total MACSec Sessions : 2
```

```
     Secured Sessions : 1
     Pending Sessions : 0
      Active Sessions : 1

RP/0/RSP0/CPU0:router#show macsec mka session interface Fo0/0/0/1/0
=============================================================================================
   Interface-Name        Local-TxSCI       #Peers   Status   Key-Server   PSK/EAP      CKN
=============================================================================================
   Fo0/0/0/1/0        d46d.5023.3709/0001     1      Secured      YES      PRIMARY     5555
   Fo0/0/0/1/0        d46d.5023.3709/0001     1      Active       YES     FALLBACK     5556
```

The **Status** field in the output confirms that the respective interface is **Secured**. If MACsec encryption is not successfully configured, you will see a status such as **Pending** or **Init**.

Note In the VPLS network, because of the configuration on a multi-point interface, the number of live peers displayed is more than 1.

Run the **show run macsec-policy** command in the privileged executive mode to troubleshoot the configuration entered.

Step 3 Verify whether the interface of the router is peering with its neighbor after MACsec configuration

Example:

```
RP/0/RSP0/CPU0:router#show macsec mka session

NODE: node0_0_CPU0

=================================================================
Interface    Local-TxSCI           # Peers   Status  Key-Server
=================================================================

Fo0/0/0/1/0  001d.e5e9.aa39/0005      1      Secured  YES
```

The following is a sample output that displays active fallback PSK entry as well:

```
Router#show macsec mka session
Wed Apr 28 01:59:39.478 UTC

NODE: node0_1_CPU0
=============================================================================================
   Interface-Name        Local-TxSCI       #Peers   Status   Key-Server   PSK/EAP      CKN
=============================================================================================
     Fo0/0/0/1/0      001d.e5e9.aa39/0005     1      Secured      NO       PRIMARY     1234
     Fo0/0/0/1/0      001d.e5e9.aa39/0005     1      Active       NO       FALLBACK    1111
```

The **#Peers** field in the output confirms the presence of the peer you have configured on the physical interface, **Fo0/0/0/1/0**. If the number of peers is not reflected accurately in this output, run the **show run** command and verify the peer configuration on the interface.

Note If the MKA session status is shown as **Secured** with **0 (Zero)** peer count, this means that the link is locally secured (Tx). This is because of MKA peer loss caused by **No Rx Packets (MKA Packet)** from that peer.

Note In the VPLS network, because of the configuration on a multipoint interface, the number of live peers displayed is more than 1.

```
Router#show macsec mka session
Fri May 28 07:18:45.726 UTC

NODE: node0_0_CPU0
==============================================================================================
Interface-Name        Local-TxSCI       #Peers   Status   Key-Server   PSK/EAP      CKN
==============================================================================================
  Te0/0/0/1       6c8b.d34f.0635/0001     2      Secured      NO      FALLBACK     5556
```

Step 4 Verify whether the MKA session is secured with MACsec on the respective interface.

Example:

Before the introduction of active fallback functionality:

```
RP/0/RSP0/CPU0:router#show macsec mka session interface Fo0/0/0/1/0 detail
MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec
Local Tx-SCI            : 001d.e5e9.aa39/0005
Local Tx-SSCI           : 1
Interface MAC Address   : 001d.e5e9.aa39
MKA Port Identifier     : 1
Interface Name          : Fo0/0/0/1/0
CAK Name (CKN)          : 1020000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)  : A880BB45B9CE01584535F239
Message Number (MN)     : 5382
Authenticator           : NO
Key Server              : YES
MKA Cipher Suite        : AES-128-CMAC
Latest SAK Status       : Rx & Tx
Latest SAK AN           : 0
Latest SAK KI (KN)      : A880BB45B9CE01584535F23900000001 (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         : scale-21
Key Server Priority     : 20
Replay Window Size      : 40
Confidentiality Offset  : 50
Algorithm Agility       : 80C201
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 Peer List:
  MI                        MN          Rx-SCI (Peer)        SSCI KS-Priority
  ----------------------------------------------------------------------------
  4E33A276E7F79C04D80FE346       27114  d46d.5023.3704/0001     2         235
Potential Peer List:
  MI                        MN          Rx-SCI (Peer)        SSCI KS-Priority
  ----------------------------------------------------------------------------
```

```
! If sub-interfaces are configured, the output would be as follows:

RP/0/RSP0/CPU0:router# show macsec mka session interface Fo0/0/0/1/1.8 detail
MKA Detailed Status for MKA Session
==================================
Status: SECURED - Secured MKA Session with MACsec
Local Tx-SCI            : e0ac.f172.4124/001d
Local Tx-SSCI           : 1
Interface MAC Address   : e0ac.f172.4124
MKA Port Identifier     : 29
Interface Name          : Fo0/0/0/1/1.8
CAK Name (CKN)          : ABC1000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)  : 1EC4A4D1B0D75D3D5C2F6393
Message Number (MN)     : 1915
Authenticator           : NO
Key Server              : NO
MKA Cipher Suite        : AES-128-CMAC
Latest SAK Status       : Rx & Tx
Latest SAK AN           : 3
Latest SAK KI (KN)      : EB1E04894327E4EFA283C66200000003 (3)
Old SAK Status          : No Rx, No Tx
Old SAK AN              : 0
Old SAK KI (KN)         : RETIRED (4)
SAK Transmit Wait Time  : 0s (Not waiting for any peers to respond)
SAK Retire Time         : 0s (No Old SAK to retire)
MKA Policy Name         : test12
Key Server Priority     : 0
Replay Window Size      : 1024
Confidentiality Offset  : 50
Algorithm Agility       : 80C201
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 : 0
Live Peer List:
  MI                      MN          Rx-SCI (Peer)        SSCI KS-Priority
  ----------------------------------------------------------------------------
  EB1E04894327E4EFA283C662        1908  001d.e5e9.b1c0/0037     2           0
Potential Peer List:
  MI                      MN          Rx-SCI (Peer)        SSCI KS-Priority
  ----------------------------------------------------------------------------
RP/0/RSP0/CPU0:macsec-CE1#sh macsec mka  interface Fo0/0/0/1/1.8
============================================================
Interface-name          KeyChain-Name           Policy Name
============================================================
Fo0/0/0/1/1.8           kc3                     test12


! In a VPLS network with multipoint interface, the output would be as follows:

RP/0/RSP0/CPU0:router#show macsec mka session interface FortyGigE0/0/0/1/0.1 detail
MKA Detailed Status for MKA Session
==================================
Status: SECURED - Secured MKA Session with MACsec
Local Tx-SCI            : e0ac.f172.4123/0001
Local Tx-SSCI           : 1
Interface MAC Address   : e0ac.f172.4123
MKA Port Identifier     : 1
Interface Name          : Fo0/0/0/1/0.1
CAK Name (CKN)          : ABC1000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)  : A1DB3E42B4A543FBDBC281A6
```

```
Message Number (MN)     : 1589
Authenticator           : NO
Key Server              : NO
MKA Cipher Suite        : AES-128-CMAC
Latest SAK Status       : Rx & Tx
Latest SAK AN           : 1
Latest SAK KI (KN)      : AEC899297F5B0BDEF7C9FC6700000002 (2)
Old SAK Status          : No Rx, No Tx
Old SAK AN              : 0
Old SAK KI (KN)         : RETIRED (1)
SAK Transmit Wait Time  : 0s (Not waiting for any peers to respond)
SAK Retire Time         : 0s (No Old SAK to retire)
MKA Policy Name         : mk_xpn1
Key Server Priority     : 0
Replay Window Size      : 1024
Confidentiality Offset  : 50
Algorithm Agility       : 80C201
SAK Cipher Suite        : 0080C20001000004 (GCM-AES-XPN-256)
MACsec Capability       : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired          : YES
# of MACsec Capable Live Peers           : 2
# of MACsec Capable Live Peers Responded : 0
Live Peer List:
  MI                        MN          Rx-SCI (Peer)        SSCI KS-Priority
  -----------------------------------------------------------------------------
  AEC899297F5B0BDEF7C9FC67         225  001d.e5e9.b1bf/0001     3           0
  0A4C49EE5B7401F1BECB7E22         147  001d.e5e9.f329/0001     2           0
Potential Peer List:
  MI                        MN          Rx-SCI (Peer)        SSCI KS-Priority
  -----------------------------------------------------------------------------
```

With the introduction of active fallback functionality:

The following show command output verifies that the primary and fallback keys (CAK) are matched on both peer ends.

```
RP/0/RSP0/CPU0:router#show macsec mka session interface Hu0/0/0/11 detail

MKA Detailed Status for MKA Session
===================================
Status: Secured - Secured MKA Session with MACsec

Local Tx-SCI                    : 7061.7bea.1df4/0001
Local Tx-SSCI                   : 1
Interface MAC Address           : 7061.7bea.1df4
MKA Port Identifier             : 1
Interface Name                  : Hu0/0/0/11
CAK Name (CKN)                  : 2111
CA Authentication Mode          : PRIMARY-PSK
Keychain                        : test1
Member Identifier (MI)          : 42A78BD6243539E917B8C6B2
Message Number (MN)             : 555
Authenticator                   : NO
Key Server                      : NO
MKA Cipher Suite                : AES-128-CMAC
Configured MACSec Cipher Suite  : GCM-AES-XPN-128

Latest SAK Status               : Rx & Tx
Latest SAK AN                   : 0
Latest SAK KI (KN)              : 69B39E87B3CBA673401E989100000001 (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)
Time to SAK Rekey               : NA
Time to exit suspension         : NA

MKA Policy Name                 : P12
Key Server Priority             : 20
Delay Protection                : TRUE
Replay Window Size              : 100
Include ICV Indicator           : TRUE
Confidentiality Offset          : 0
Algorithm Agility               : 80C201
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 : 0

Live Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------
69B39E87B3CBA673401E9891      617       008a.96d6.194c/0001    2        20

Potential Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------

Peers Status:
Last Tx MKPDU          : 2021 May 18 13:27:56.548
Peer Count             : 1

RxSCI                  : 008A96D6194C0001
  MI                    : 69B39E87B3CBA673401E9891
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 18 13:27:56.518

MKA Detailed Status for MKA Session
===================================
Status: Active - Marked Peer as Live (Waiting for SAK generation/distribution)

Local Tx-SCI                    : 7061.7bea.1df4/0001
Local Tx-SSCI                   : 1
Interface MAC Address           : 7061.7bea.1df4
MKA Port Identifier             : 1
Interface Name                  : Hu0/0/0/11
CAK Name (CKN)                  : 2000
CA Authentication Mode          : FALLBACK-PSK
Keychain                        : test1f
Member Identifier (MI)          : 1BB9428C721F6EE3E538C942
Message Number (MN)             : 553
Authenticator                   : NO
Key Server                      : NO
MKA Cipher Suite                : AES-128-CMAC
Configured MACSec Cipher Suite  : GCM-AES-XPN-128

Latest SAK Status               : Rx & Tx
Latest SAK AN                   : 0
Latest SAK KI (KN)              : 69B39E87B3CBA673401E989100000001 (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)
Time to SAK Rekey               : NA
Time to exit suspension         : NA

MKA Policy Name                 : P12
Key Server Priority             : 20
Delay Protection                : TRUE
Replay Window Size              : 100
Include ICV Indicator           : TRUE
Confidentiality Offset          : 0
Algorithm Agility               : 80C201
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 : 0

Live Peer List:
------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
------------------------------------------------------------------------------
8F59AD6021FA3E2D5F9E6231      615       008a.96d6.194c/0001    2        20

Potential Peer List:
------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
------------------------------------------------------------------------------

Peers Status:
Last Tx MKPDU          : 2021 May 18 13:27:56.547
Peer Count             : 1

RxSCI                  : 008A96D6194C0001
  MI                   : 8F59AD6021FA3E2D5F9E6231
  Peer CAK             : Match
  Latest Rx MKPDU      : 2021 May 18 13:27:56.518

RP/0/RSP0/CPU0:router#
```

If sub-interfaces are configured, the output would be as follows. In this example, the status of FALLBACK-PSK is *Secured*.

```
RP/0/RSP0/CPU0:router# show macsec mka session interface Hu0/0/0/0.6 detail
MKA Detailed Status for MKA Session
===================================
Status: Secured - Secured MKA Session with MACsec

Local Tx-SCI                    : 7061.7bea.1dc8/0006
Local Tx-SSCI                   : 1
Interface MAC Address           : 7061.7bea.1dc8
MKA Port Identifier             : 6
Interface Name                  : Hu0/0/0/0.6
CAK Name (CKN)                  : 9999
CA Authentication Mode          : FALLBACK-PSK
Keychain                        : D_tagf
Member Identifier (MI)          : 1DE18714A098B80964CC651E
Message Number (MN)             : 6203
Authenticator                   : NO
Key Server                      : YES
MKA Cipher Suite                : AES-128-CMAC
```

```
Configured MACSec Cipher Suite  : GCM-AES-XPN-256

Latest SAK Status               : Rx & Tx
Latest SAK AN                   : 0
Latest SAK KI (KN)              : 1DE18714A098B80964CC651E00000001 (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)
Time to SAK Rekey               : 23510s
Time to exit suspension         : NA

MKA Policy Name                 : D_tag1
Key Server Priority             : 1
Delay Protection                : FALSE
Replay Window Size              : 1000
Include ICV Indicator           : TRUE
Confidentiality Offset          : 50
Algorithm Agility               : 80C201
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

# of MACSec Suspended Peers              : 0

Live Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------
5C852D8F920306893D2BFB8F     10978      00c1.645f.2dd4/0006    2        11

Potential Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------

Suspended Peer List:
--------------------------------------------------------------------------------
       Rx-SCI          SSCI
--------------------------------------------------------------------------------

Peers Status:
Last Tx MKPDU          : 2021 May 18 13:29:15.687
Peer Count             : 1

RxSCI                  : 00C1645F2DD40006
  MI                    : 5C852D8F920306893D2BFB8F
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 18 13:29:15.769

RP/0/RSP0/CPU0:router#


! In a VPLS network with multipoint interface, the output would be as follows:

RP/0/RSP0/CPU0:router#show macsec mka session interface Hu0/0/1/7 detail
Fri May 28 07:19:11.362 UTC

MKA Detailed Status for MKA Session
```

```
==================================
Status: Secured - Secured MKA Session with MACsec

Local Tx-SCI                    : 6c8b.d34f.0635/0001
Local Tx-SSCI                   : 2
Interface MAC Address           : 6c8b.d34f.0635
MKA Port Identifier             : 1
Interface Name                  : Te0/0/0/1
CAK Name (CKN)                  : 5556
CA Authentication Mode          : FALLBACK-PSK
Keychain                        : test2f
Member Identifier (MI)          : 6D14ECCDFB70E7E0463BD509
Message Number (MN)             : 20455
Authenticator                   : NO
Key Server                      : NO
MKA Cipher Suite                : AES-256-CMAC
Configured MACSec Cipher Suite  : GCM-AES-XPN-256

Latest SAK Status               : Rx & Tx
Latest SAK AN                   : 2
Latest SAK KI (KN)              : 1BBDDC0520C797C26AB7F1BF00000002 (2)
Old SAK Status                  : No Rx, No Tx
Old SAK AN                      : 1
Old SAK KI (KN)                 : RETIRED (1)

SAK Transmit Wait Time          : 0s (Not waiting for any peers to respond)
SAK Retire Time                 : 0s (No Old SAK to retire)
Time to SAK Rekey               : NA
Time to exit suspension         : NA

MKA Policy Name                 : *DEFAULT POLICY*
Key Server Priority             : 16
Delay Protection                : FALSE
Replay Window Size              : 64
Include ICV Indicator           : FALSE
Confidentiality Offset          : 0
Algorithm Agility               : 80C201
SAK Cipher Suite                : 0080C20001000004 (GCM-AES-XPN-256)
MACsec Capability               : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired                  : YES

# of MACsec Capable Live Peers           : 2
# of MACsec Capable Live Peers Responded : 0

Live Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------
1BBDDC0520C797C26AB7F1BF     19997      008a.96d6.194c/0001    3        16
B25B1000CC6FAE92D1F85738      139       dc77.4c3e.59c3/0001    1        16

Potential Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------

Peers Status:
 Last Tx MKPDU          : 2021 May 28 07:19:10.153
 Peer Count             : 2

 RxSCI                  : 008A96D6194C0001
  MI                    : 1BBDDC0520C797C26AB7F1BF
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 28 07:19:09.960
```

```
 RxSCI                  : DC774C3E59C30001
  MI                    : B25B1000CC6FAE92D1F85738
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 28 07:19:10.180

RP/0/RSP0/CPU0:router#


RP/0/RSP0/CPU0:router#show macsec mka session interface Hu0/0/1/7.1 detail

MKA Detailed Status for MKA Session
===================================
Status: Secured - Secured MKA Session with MACsec

Local Tx-SCI                    : 7061.7bff.e5e8/0001
Local Tx-SSCI                   : 2
Interface MAC Address           : 7061.7bff.e5e8
MKA Port Identifier             : 1
Interface Name                  : Hu0/0/1/7.1
CAK Name (CKN)                  : 5556
CA Authentication Mode          : FALLBACK-PSK
Keychain                        : test22f
Member Identifier (MI)          : 8FF3D1BBF09EA4AD6A0FC1B5
Message Number (MN)             : 81
Authenticator                   : NO
Key Server                      : YES
MKA Cipher Suite                : AES-256-CMAC
Configured MACSec Cipher Suite  : GCM-AES-XPN-256

Latest SAK Status               : Rx & Tx
Latest SAK AN                   : 3
Latest SAK KI (KN)              : 8FF3D1BBF09EA4AD6A0FC1B500000002 (2)
Old SAK Status                  : No Rx, No Tx
Old SAK AN                      : 2
Old SAK KI (KN)                 : RETIRED (1)

SAK Transmit Wait Time          : 0s (Not waiting for any peers to respond)
SAK Retire Time                 : 0s (No Old SAK to retire)
Time to SAK Rekey               : 17930s
Time to exit suspension         : NA

MKA Policy Name                 : P123
Key Server Priority             : 10
Delay Protection                : FALSE
Replay Window Size              : 64
Include ICV Indicator           : FALSE
Confidentiality Offset          : 30
Algorithm Agility               : 80C201
SAK Cipher Suite                : 0080C20001000004 (GCM-AES-XPN-256)
MACsec Capability               : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired                  : YES

# of MACsec Capable Live Peers           : 2
# of MACsec Capable Live Peers Responded : 2

# of MACSec Suspended Peers              : 0

Live Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------
6BCF91135F807CB9F57DDAAA       61       dc77.4c3e.5b05/0001    1        24
D81CFE93D07E932DDC33666E       44       00a7.4250.56c2/0001    3        25
```

```
Potential Peer List:
-----------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
-----------------------------------------------------------------------------

Suspended Peer List:
-----------------------------------------------------------------------------
       Rx-SCI          SSCI
-----------------------------------------------------------------------------

Peers Status:
 Last Tx MKPDU          : 2021 May 28 13:16:50.992
 Peer Count             : 2

 RxSCI                  : DC774C3E5B050001
  MI                    : 6BCF91135F807CB9F57DDAAA
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 28 13:16:51.312

 RxSCI                  : 00A7425056C20001
  MI                    : D81CFE93D07E932DDC33666E
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 28 13:16:50.945
RP/0/RSP0/CPU0:router#
```

Step 5 Verify the MACsec session counter statistics.

Example:

```
RP/0/RSP0/CPU0:router# show macsec mka statistics interface Fo0/0/0/1/0


MKA Statistics for Session on interface (Fo0/0/0/1/0)
=====================================================
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.......... 3
SAKs Rekeyed............ 2
SAKs Received........... 0
SAK Responses Received.. 3

MKPDU Statistics
MKPDUs Transmitted...... 5425
"Distributed SAK".. 8
"Distributed CAK".. 0
MKPDUs Validated & Rx... 4932
"Distributed SAK".. 0
"Distributed CAK".. 0

MKA IDB Statistics
MKPDUs Tx Success.......... 5425
MKPDUs Tx Fail............. 0
MKPDUS Tx Pkt build fail... 0
MKPDUs Rx CA Not found..... 0
```

```
MKPDUs Rx Error..... 0
MKPDUs Rx Success..... 4932

MKPDU Failures
   MKPDU Rx Validation (ICV)..... 0
   MKPDU Rx Bad Peer MN..... 0
   MKPDU Rx Non-recent Peerlist MN..... 0
   MKPDU Rx Drop SAKUSE, KN mismatch..... 0
   MKPDU Rx Drop SAKUSE, Rx Not Set..... 0
   MKPDU Rx Drop SAKUSE, Key MI mismatch.. 0
   MKPDU Rx Drop SAKUSE, AN Not in Use.... 0
   MKPDU Rx Drop SAKUSE, KS Rx/Tx Not Set. 0

SAK Failures
   SAK Generation..... 0
   Hash Key Generation..... 0
   SAK Encryption/Wrap..... 0
   SAK Decryption/Unwrap..... 0
```

! If sub-interfaces are configured, the output would be as follows:

```
RP/0/RSP0/CPU0:router# show macsec mka statistics interface Fo0/0/0/1/1.8

MKA Statistics for Session on interface (Fo0/0/0/1/1.8)
=======================================================
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..... 0
   SAKs Rekeyed..... 0
   SAKs Received..... 9
   SAK Responses Received.. 0
MKPDU Statistics
   MKPDUs Transmitted..... 1973
      "Distributed SAK".. 0
      "Distributed CAK".. 0
   MKPDUs Validated & Rx... 1965
      "Distributed SAK".. 9
      "Distributed CAK".. 0
MKA IDB Statistics
   MKPDUs Tx Success..... 1973
   MKPDUs Tx Fail..... 0
   MKPDUS Tx Pkt build fail... 0
   MKPDUs Rx CA Not found..... 0
   MKPDUs Rx Error..... 0
   MKPDUs Rx Success..... 1965
```

! In a VPLS network with a mulitpoint interface, the output would be as follows:

```
RP/0/RSP0/CPU0:router# show macsec mka statistics interface FortyGigE0/0/0/1/0.1

MKA Statistics for Session on interface (Fo0/0/0/1/0.1)
=======================================================
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.......... 0
   SAKs Rekeyed............ 0
   SAKs Received........... 2
   SAK Responses Received.. 0
MKPDU Statistics
   MKPDUs Transmitted...... 1608
      "Distributed SAK".. 0
      "Distributed CAK".. 0
   MKPDUs Validated & Rx... 406
      "Distributed SAK".. 2
      "Distributed CAK".. 0
MKA IDB Statistics
   MKPDUs Tx Success.......... 1608
   MKPDUs Tx Fail............. 0
   MKPDUS Tx Pkt build fail... 0
   MKPDUs Rx CA Not found..... 0
   MKPDUs Rx Error............ 0
   MKPDUs Rx Success.......... 1802
```

The counters display the MACsec PDUs transmitted, validated, and received. The output also displays transmission errors, if any.

This completes the verification of MACsec encryption on the IOS-XR.

Verifying MACsec Encryption on ASR 9000

MACsec encryption on the router hardware can be verified by running relevant commands in the Privileged Executive Mode.

To verify if MACsec encryption has been correctly configured, follow these steps.

SUMMARY STEPS

1. Verify the MACsec encryption and hardware interface descriptor block (IDB) information on the interface.
2. Use the IDB handle retrieved from Step 1 to verify the platform hardware information.
3. Use the Transmitter SA retrieved from Step 2 to verify the MACsec SA information programmed in the hardware.
4. Verify the MACsec Secure Channel (SC) information programmed in the hardware.

DETAILED STEPS

Procedure

Step 1 Verify the MACsec encryption and hardware interface descriptor block (IDB) information on the interface.

Example:

```
RP/0/RSP0/CPU0:router# show macsec ea idb interface Fo0/0/0/1/0


IDB Details:
if_sname : Fo0/0/0/1/0
```

```
if_handle : 0x3480
Replay window size : 64
Local MAC : 00:1d:e5:e9:aa:39
Rx SC Option(s) : Validate-Frames Replay-Protect
Tx SC Option(s) : Protect-Frames Always-Include-SCI
Security Policy : MUST SECURE
Sectag offset : 8
VLAN : Outer tag (etype=0x8100, id=1, priority=0, cfi=0): Inner tag (etype=0x8100, id=1, priority=0,
 cfi=0)
Rx SC 1
Rx SCI : 001de5e9b1bf0019
Peer MAC : 00:1d:e5:e9:b1:bf
Stale : NO
SAK Data
SAK[0] : ***
SAK Len : 32
HashKey[0] : ***
HashKey Len : 16
Conf offset : 30
Cipher Suite : GCM-AES-XPN-256
CtxSalt[0] : 83 c3 7b ad 7b 6f 63 16 09 8f f3 d2
Rx SA Program Req[0]: 2015 Oct 09 15:20:53.082
Rx SA Program Rsp[0]: 2015 Oct 09 15:20:53.092

Tx SC
Tx SCI : 001de5e9aa39001a
Active AN : 0
Old AN : 255
Next PN : 1, 0, 0, 0
SAK Data
SAK[0] : ***
SAK Len : 32
HashKey[0] : ***
HashKey Len : 16
Conf offset : 30
Cipher Suite : GCM-AES-XPN-256
CtxSalt[0] : 83 c3 7b ae 7b 6f 63 16 09 8f f3 d2
Tx SA Program Req[0]: 2015 Oct 09 15:20:55.053
Tx SA Program Rsp[0]: 2015 Oct 09 15:20:55.064
```

! When more than 1 RX SA is configured in P2MP networks, the output would be as follows:

```
RP/0/RSP0/CPU0:router# show macsec ea idb interface FortyGigE0/0/0/1/0.1
IDB Details:
  if_sname                : Fo0/0/0/1/0.1
  if_handle               : 0x2e40
  Replay window size      : 1024
  Local MAC               : e0:ac:f1:72:41:23
  Rx SC Option(s)         : Validate-Frames Replay-Protect
  Tx SC Option(s)         : Protect-Frames Always-Include-SCI
  Security Policy         : MUST SECURE
  Sectag offset           : 8
  VLAN                    : Outer tag (etype=0x8100, id=1, priority=0, cfi=0)
                          : Inner tag (etype=0x8100, id=1, priority=0, cfi=0)
  Rx SC 1
    Rx SCI                : 001de5e9f3290001
    Peer MAC              : 00:1d:e5:e9:f3:29
    Stale                 : NO
    SAK Data
      SAK[1]              : ***

      SAK Len             : 32
      HashKey[1]          : ***
      HashKey Len         : 16
```

```
    Conf offset         : 50
    Cipher Suite        : GCM-AES-XPN-256
    CtxSalt[1]          : ae ca 99 2b 7f 5b 0b de f7 c9 fc 67
  Rx SC 2
    Rx SCI              : 001de5e9b1bf0001
    Peer MAC            : 00:1d:e5:e9:b1:bf
    Stale               : NO
    SAK Data
      SAK[1]            : ***

      SAK Len           : 32
      HashKey[1]        : ***
      HashKey Len       : 16
      Conf offset       : 50
      Cipher Suite      : GCM-AES-XPN-256
      CtxSalt[1]        : ae ca 99 2a 7f 5b 0b de f7 c9 fc 67
  Tx SC
    Tx SCI              : e0acf17241230001
    Active AN           : 1
    Old AN              : 0
    Next PN             : 1, 1, 0, 0
    SAK Data
      SAK[1]            : ***

      SAK Len           : 32
      HashKey[1]        : ***
      HashKey Len       : 16
      Conf offset       : 50
      Cipher Suite      : GCM-AES-XPN-256
      CtxSalt[1]        : ae ca 99 28 7f 5b 0b de f7 c9 fc 67
```

The **if_handle** field provides the IDB instance location.

The **Replay window size** field displays the configured window size.

The **Security Policy** field displays the configured security policy.

The **Local Mac** field displays the MAC address of the router.

The **Peer Mac** field displays the MAC address of the peer. This confirms that a peer relationship has been formed between the two routers.

Step 2 Use the IDB handle retrieved from Step 1 to verify the platform hardware information.

Example:

```
RP/0/RSP0/CPU0:router# show macsec ea platform hardware
idb location 0/0/CPU0 | b 3480

if_handle : 0x00003480
NPPort : 099 [0x063]
LdaPort : 016 [0x010] SerdesPort : 000 [0x000]
NetSoftPort : 061 [0x03d] SysSoftPort : 062 [0x03e]
Active AN : 0x00000000 Idle AN : 0x000000ff
Match-All Tx SA : 0x80010001 Match-All Rx SA : 0x00010001
Match-All Tx Flow : 0x80000003 Match-All Rx Flow : 0x00000003
Bypass Tx SA : 0x80000000 Bypass Rx SA : 0x00000000
Tx SA[0] : 0x80020002 Tx Flow[0] : 0x8000000c
Tx SA[1] : 0xffffffff Tx Flow[1] : 0xffffffff
Tx SA[2] : 0xffffffff Tx Flow[2] : 0xffffffff
Tx SA[3] : 0xffffffff Tx Flow[3] : 0xffffffff
Rx SA[0] : 0x00020002 Rx Flow[0] : 0x0000000c
Rx SA[1] : 0xffffffff Rx Flow[1] : 0xffffffff
```

```
Rx SA[2] : 0xffffffff Rx Flow[2] : 0xffffffff
Rx SA[3] : 0xffffffff Rx Flow[3] : 0xffffffff
```

Step 3 Use the Transmitter SA retrieved from Step 2 to verify the MACsec SA information programmed in the hardware.

Example:

```
RP/0/RSP0/CPU0:router# show macsec ea platform hardware sa
0x80020002 interface Fo0/0/0/1/0 location 0/0/CPU0

MACSEC HW SA Details:
Action Type : 0x00000003
Direction : Egress
Dest Port : 0x00000000
Conf Offset : 00000030
Drop Type : 0x00000002
Drop NonResvd : 0x00000000
SA In Use : YES
ConfProtect : YES
IncludeSCI : YES
ProtectFrame : YES
UseEs : NO
UseSCB : NO
SCI : 00 1d e5 e9 aa 39 00 05
Replay Window : 64 MacsecCryptoAlgo : 7
Direction : Egress AN : 0
AES Key Len : 256 X-Packet Number : 0x0000000000000000
CtxSalt : f8d88dc3e1c5e6a94ca2299
```

The output displays the details of the encryption, such as the AES key, the Auth key, and other parameters.

Step 4 Verify the MACsec Secure Channel (SC) information programmed in the hardware.

Example:

```
RP/0/RSP0/CPU0:router# show macsec ea platform hardware msc
interface Fo0/0/0/1/0 location 0/0/CPU0


MACSEC HW Cfg Details:
Mode : 0x5
Counter Clear on Read : 0x0
SA Fail Mask : 0xffff
VlanCounter Update : 0x1
Global SecFail Mask : 0xffffffff
Latency : 0xff
StaticBypass : 0x0
Should secure : 0x0
Global Frame Validation : 0x2
Ctrl Pkt CC Bypass : 0x1
NonCtrl Pkt CC Bypass : 0x1
Sequence Number Threshold : 0xbffffb8
Sequence Number Threshold 64bit : 0x000002fffffffffd
Non Matching Non Control Pkts Programming
      Untagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
      Tagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
      BadTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
      KayTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
Non Matching Control Pkts Programming
```

```
Untagged : Bypass: 0x1 DestPort : 0x2, DropType : 0xffffffff
Tagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
BadTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
KayTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
```

This completes the verification of MACsec encryption on the router hardware.

This completes the configuration and verification of MACsec encryption.

Configuring and Verifying MACsec Encryption as a Service

This section describes how MACsec can be implemented as a service in a L2VPN or L3VPN setup.

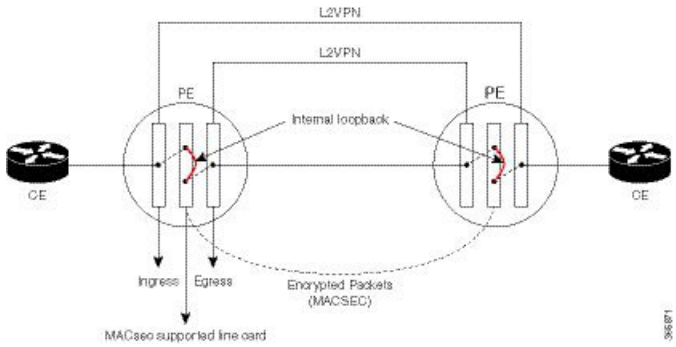
Note MACsec encryption is not supported on interface bundles, but is supported on member links .

Use Case 1: MACsec in an L2VPN Topology

In this topology, MACsec is configured on the PE router (with the interfaces facing the CE router) to provide crypto or encryption service on the PE router as a premium service for selected traffic on the WAN core. The interfaces can be physical ethernet interfaces or VLAN sub-interfaces. The customer can select the traffic that will be part of the encryption.

The following figure illustrates the use of MACsec as a service in an L2VPN network:

Figure 13: MACsec in an L2VPN topology



The data transferred between the CE router and the PE router are not encrypted. The data in clear format is sent to the access port of the PE router.

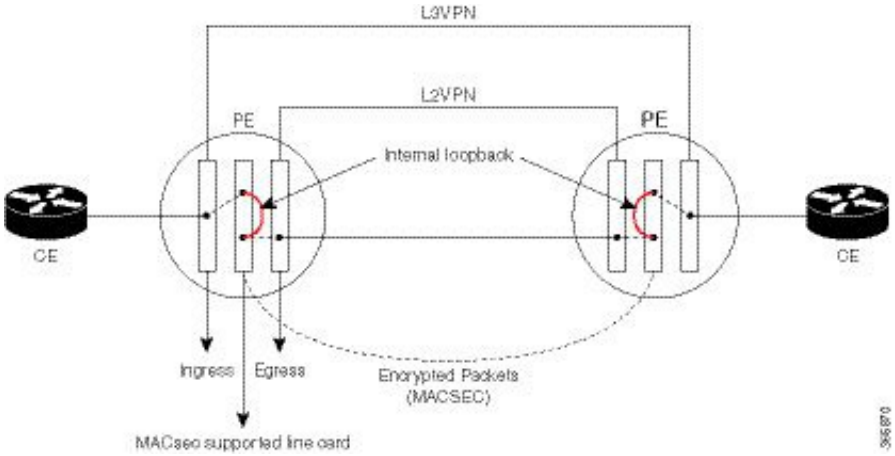
The PE router ports that receive traffic from CE routers divert the traffic using L2 local switching to the line card configured to perform encryption. The MACsec configuration creates internal loopback to the port configured for L2VPN to the opposite PE. After this, the packets are sent completely encrypted to the opposite PE router.

Use Case 2: MACsec in an L3VPN Topology

The following figure illustrates the use of MACsec as a service in an L3VPN environment. The topology is similar to an L2VPN set up where MACsec is configured on the PE router (where the interfaces facing the

CE router) to provide crypto or encryption services on the PE router as a premium service for selected traffic on the WAN core.

Figure 14:



The data transferred between the CE router and the PE router is not encrypted. The data is sent in clear-text format to the PE router access port. The PE router for each sub-interface distinguishes whether the data is part of MACsec encrypted service.

The PE router ports that receive traffic from CE routers divert the traffic using L3 local switching to the line card port configured to do encryption. The MACsec configuration creates internal loopback to the port configured for L2VPN to the opposite PE router. After this, the packets are sent completely encrypted to the opposite PE.

Restrictions

Ports usage for encryption on the line card must meet the following criteria:

- The ports must be TenGigE interfaces.
- Both the ports must belong either to an A9K-MPA-20X10GE MPA, or they must be breakout interfaces from one of the A9K-8X100GE-SE, A9K-8X100GE-TR, A9K-4X100GE-SE, or A9K-4X100GE-TR line cards.
- If the interfaces belong to A9K-MPA-20x10GE line card, then both the interfaces must be either in port range 0-9, or in port range 10-19. One interface from range 0-9 and other from 10-19 must not be selected.
- If the interfaces are breakout interfaces, then both of them must belong to the same HundredGigE port.

Note These restrictions apply only to MACsec interfaces. These restrictions do not apply to the CE or core-facing interfaces.

Configuring MACsec as a Service

SUMMARY STEPS

1. Enter interface configuration mode.
2. Configure the MACsec service.
3. Commit your configuration and exit global configuration mode.
4. Confirm the MACsec policy configuration.

DETAILED STEPS

Procedure

Step 1 Enter interface configuration mode.

Example:

```
RP/0/RSP/CPU0:router# interface <interface> 15.10 l2transport
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 10
```

Step 2 Configure the MACsec service.

Example:

```
RP/0/RSP0/CPU0:router(config-subif)# macsec-service decrypt-port <intf>17.10 psk-keychain
<keychain_name> [policy <macsec_policy>]
```

Step 3 Commit your configuration and exit global configuration mode.

Example:

```
RP/0/RSP0/CPU0:router# commit
RP/0/RSP0/CPU0:router# exit
```

Step 4 Confirm the MACsec policy configuration.

Example:

```
RP/0/RSP0/CPU0:router#

show running-config interface <interface> 15.10

interface <interface> 15.10
 macsec-service decrypt-port <intf>17.10 psk-keychain <keychain_name> [policy <macsec_policy>]
 encapsulation dot1q 10
```

Configuring MACsec Service for L2VPN Network

Configuring the MACsec service for L2VPN network, involves the following steps:

SUMMARY STEPS

1. Enter global configuration mode.
2. Enter interface configuration mode and configure port facing the CE router.
3. Enable MACsec service.
4. Configure service port.
5. Configure the Xconnect group between ports.
6. Connect the ports.

DETAILED STEPS

Procedure

Step 1 Enter global configuration mode.

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Step 2 Enter interface configuration mode and configure port facing the CE router.

The interface can be a physical interface or a VLAN sub-interface.

Example:

```
RP/0/RSP0/CPU0:router(config)# interface  <interface>15.10  l2transport
 encapsulation dot1q 10
```

Step 3 Enable MACsec service.

Example:

```
RP/0/RSP0/CPU0:router(config-if)# interface  <interface>16.10  l2transport
 encapsulation dot1q 10
  macsec-service decrypt-port <intf>17.10 psk-keychain <keychain_name> [policy <macsec_policy>]
```

Step 4 Configure service port.

Example:

```
RP/0/RSP0/CPU0:router(config-if)# interface  <interface>17.10  l2transport
 encapsulation dot1q 10
```

Step 5 Configure the Xconnect group between ports.

Example:

```
RP/0/RSP0/CPU0:router(config-if)# l2vpn
xconnect group local_macsec
```

```
  p2p local_macsec
   interface  <interface>15.10
   interface  <interface>16.10
```

Step 6 Connect the ports.

Example:

```
RP/0/RSP0/CPU0:router(config-if)l2vpn
 xconnect group ext_macsec
  p2p ext_macsec
   interface  <interface>17.10
   neighbor  ipv4  <a.b.c.d>  pw-id  <num>
!
```

Configuring MACsec Service for L3VPN Network

Configuring the MACsec service for L3VPN network, involves the following steps:

SUMMARY STEPS

1. Enter global configuration mode.
2. Enter interface configuration mode and configure port facing the CE router
3. Configure the PE1 router with virtual routing details.
4. Enable MACsec service.
5. Configure service port.
6. Configure the Xconnect between ports.
7. Configure ports.
8. Configure OSPF on the core interface.
9. Configure MPLS on the core interface.

DETAILED STEPS

Procedure

Step 1 Enter global configuration mode.

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Step 2 Enter interface configuration mode and configure port facing the CE router

Example:

```
RP/0/RSP0/CPU0:router(config-if)# interface TenGigE0/4/0/0.1
ipv4 address 161.1.1.1 255.255.255.0
encapsulation dot1q 1
```

Step 3 Configure the PE1 router with virtual routing details.

Example:

```
RP/0/RSP0/CPU0:router(config-if)# interface TenGigE0/3/0/0/1.1
 vrf vrf_1
 ipv4 address 161.1.1.2 255.255.255.0
 encapsulation dot1q 1
```

Step 4 Enable MACsec service.

Example:

```
RP/0/RSP0/CPU0:router(config-if)# interface TenGigE0/3/0/0/2.1
 vrf vrf_1
 ipv4 address 181.1.1.1 255.255.255.0
 macsec-service decrypt-port TenGigE0/3/0/0/3.1 psk-keychain script_key_chain1
 encapsulation dot1q 1
```

Step 5 Configure service port.

Example:

```
RP/0/RSP0/CPU0:router(config-if)#interface TenGigE0/3/0/0/3.1 l2transport
 encapsulation dot1q 1
!
```

Step 6 Configure the Xconnect between ports.

Example:

```
RP/0/RSP0/CPU0:router#configure
RP/0/RSP0/CPU0:router(config)#l2vpn
 xconnect group l3serv_xc_gp_1
  p2p l3serv_xc_p2p_1
   interface TenGigE0/3/0/0/3.1
   neighbor ipv4 3.3.3.3 pw-id 1
   !
  !
```

Step 7 Configure ports.

Example:

```
RP/0/RSP0/CPU0:router#(config)
router bgp 100
 bgp router-id 2.2.2.2
 address-family ipv4 unicast
 !
 address-family vpnv4 unicast
 !
 neighbor 3.3.3.3
  remote-as 100
  update-source Loopback1
  address-family vpnv4 unicast
  !
 !
 vrf vrf_1
  rd 1234:1
  address-family ipv4 unicast
```

```
  redistribute connected
  redistribute static
 !
 neighbor 181.1.1.2
  remote-as 100
  address-family ipv4 unicast
  !
 !
!
```

Step 8 Configure OSPF on the core interface.

Example:

```
RP/0/RSP0/CPU0:router#
macsec-PE1#sh run router ospf
router ospf core
 router-id 2.2.2.2
 redistribute connected
 redistribute static
 area 0
  interface Loopback1
  !
  interface TenGigE0/1/0/1
  !
 !
```

Step 9 Configure MPLS on the core interface.

Example:

```
RP/0/RSP0/CPU0:router#
mpls ldp
 graceful-restart
 router-id 2.2.2.2
 interface TenGigE0/1/0/1
 !
!
```

Applying MACsec Service Configuration on an Interface

The MACsec service configuration is applied to the host-facing interface of a CE router.

SUMMARY STEPS

1. Enter the global configuration mode.
2. Enter the interface configuration mode.
3. If you are configuring VLAN sub-interfaces, configure the encapsulation as shown.
4. Apply the MACsec service configuration on an interface.
5. Commit your configuration.

DETAILED STEPS

Procedure

Step 1 Enter the global configuration mode.

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Step 2 Enter the interface configuration mode.

The interface can be a physical interface or a VLAN sub-interface.

Example:

```
RP/0/RSP0/CPU0:router(config)# interface Te0/3/0/1/4
```

Step 3 If you are configuring VLAN sub-interfaces, configure the encapsulation as shown.

Example:

```
! For 802.1q encapsulation with a single tag
RP/0/RSP0/CPU0:router(config-if)# encapsulation dot1q 5

! For 802.1q encapsulation with double tags
RP/0/RSP0/CPU0:router(config-if)# encapsulation dot1q 3 second-dot1q 4

! For 802.1ad encapsulation with a single tag
RP/0/RSP0/CPU0:router(config-if)# encapsulation dot1ad 5

! For 802.1ad encapsulation with double tags
RP/0/RSP0/CPU0:router(config-if)# encapsulation dot1ad 3 dot1ad 4
```

Step 4 Apply the MACsec service configuration on an interface.

To apply MACsec service configuration on an interface, use the following configuration.

Example:

```
RP/0/RSP0/CPU0:router(config-if)# macsec-service decrypt-port TenGigE0/3/0/1/5 psk-keychain
script_key_chain1 policy mk_xpn_1tag
RP/0/RSP0/CPU0:router(config-if)# exit
```

Step 5 Commit your configuration.

Example:

```
RP/0/RSP0/CPU0:router(config)# commit
```

Verifying MACsec Encryption on IOS XR

MACsec encryption on IOS XR can be verified by running relevant commands in the Privileged Executive Mode. The verification steps are the same for MACsec encryption on L2VPN or L3VPN network.

Note With the introduction of active fallback functionality in Cisco IOS XR Software Release 7.1.2 (Release 6.7.2 for 32-bit Cisco IOS XR platforms), the output of various MACsec show commands include the fallback PSK entry as well.

To verify if MACsec encryption has been correctly configured, follow these steps.

SUMMARY STEPS

1. Verify the MACsec policy configuration.
2. Verify the MACsec configuration on the respective interface.
3. Verify whether the interface of the router is peering with its neighbor after MACsec configuration
4. Verify whether the MKA session is secured with MACsec on the respective interface.
5. Verify the MACsec session counter statistics.

DETAILED STEPS

Procedure

Step 1 Verify the MACsec policy configuration.

Example:

```
RP/0/RSP0/CPU0:router#show macsec policy mac_policy

================================================================================
Policy      Cipher         Key-Svr     Window  Conf
name        Suite          Priority    Size    Offset
================================================================================
mac_policy GCM-AES-XPN-256  0          64      30
```

If the values you see are different from the ones you configured, then check your configuration by running the **show run macsec-policy** command.

Step 2 Verify the MACsec configuration on the respective interface.

You can verify the MACsec encryption on the configured interface bundle (MPLS network), P2MP interface (VPLS network), or VLAN sub-interface (EoMPLS PW network).

Example:

Before the introduction of active fallback functionality:

```
RP/0/RSP0/CPU0:router#show macsec mka summary

NODE: node0_0_CPU0
```

```
=================================================================
Interface     Status    Cipher Suite      KeyChain
=================================================================

Fo0/0/0/1/0   Secured   GCM-AES-XPN-256   mac_chain

Total MACSec Sessions : 1
     Secured Sessions : 1
     Pending Sessions : 0

RP/0/RSP0/CPU0:router#show macsec mka session interface Fo0/0/0/1/0
=========================================================================
   Interface-Name        Local-TxSCI       #Peers   Status   Key-Server
=========================================================================
   Fo0/0/0/1/0        d46d.5023.3709/0001     1     Secured      YES


! If sub-interfaces are configured, the output would be as follows:

RP/0/RSP0/CPU0:router#show macsec mka session interface Fo0/0/0/1/1.8
=========================================================================
      Interface         Local-TxSCI       # Peers   Status  Key-Server
=========================================================================
  Fo0/0/0/1/1.8      e0ac.f172.4124/001d      1     Secured     Yes
```

With the introduction of active fallback functionality:

The following is a sample output that displays active fallback PSK entry as well:

```
RP/0/RSP0/CPU0:router#show macsec mka summary

NODE: node0_0_CPU0
=============================================================================================
   Interface-Name     Status     Cipher-Suite       KeyChain       PSK/EAP      CKN
=============================================================================================
     Fo0/0/0/1/0      Secured  GCM-AES-XPN-256      mac_chain      PRIMARY      5555
     Fo0/0/0/1/0      Active   GCM-AES-XPN-256      mac_chain_fb   FALLBACK     5556

Total MACSec Sessions : 2
     Secured Sessions : 1
     Pending Sessions : 0
      Active Sessions : 1

RP/0/RSP0/CPU0:router#show macsec mka session interface Fo0/0/0/1/0
=============================================================================================
   Interface-Name        Local-TxSCI       #Peers   Status   Key-Server   PSK/EAP      CKN
=============================================================================================
   Fo0/0/0/1/0        d46d.5023.3709/0001     1      Secured      YES      PRIMARY     5555
   Fo0/0/0/1/0        d46d.5023.3709/0001     1      Active       YES     FALLBACK     5556
```

The **Status** field in the output confirms that the respective interface is **Secured**. If MACsec encryption is not successfully configured, you will see a status such as **Pending** or **Init**.

Note In the VPLS network, because of the configuration on a multi-point interface, the number of live peers displayed is more than 1.

Run the **show run macsec-policy** command in the privileged executive mode to troubleshoot the configuration entered.

Step 3 Verify whether the interface of the router is peering with its neighbor after MACsec configuration

Example:

```
RP/0/RSP0/CPU0:router#show macsec mka session

NODE: node0_0_CPU0

================================================================
Interface    Local-TxSCI           # Peers   Status  Key-Server
================================================================

Fo0/0/0/1/0  001d.e5e9.aa39/0005      1      Secured  YES
```

The following is a sample output that displays active fallback PSK entry as well:

```
Router#show macsec mka session
Wed Apr 28 01:59:39.478 UTC

NODE: node0_1_CPU0
=============================================================================================================
   Interface-Name        Local-TxSCI       #Peers   Status   Key-Server   PSK/EAP      CKN
=============================================================================================================
     Fo0/0/0/1/0      001d.e5e9.aa39/0005     1      Secured      NO       PRIMARY     1234
     Fo0/0/0/1/0      001d.e5e9.aa39/0005     1      Active       NO       FALLBACK    1111
```

The **#Peers** field in the output confirms the presence of the peer you have configured on the physical interface, **Fo0/0/0/1/0**. If the number of peers is not reflected accurately in this output, run the **show run** command and verify the peer configuration on the interface.

Note If the MKA session status is shown as **Secured** with **0 (Zero)** peer count, this means that the link is locally secured (Tx). This is because of MKA peer loss caused by **No Rx Packets (MKA Packet)** from that peer.

Note In the VPLS network, because of the configuration on a multipoint interface, the number of live peers displayed is more than 1.

```
Router#show macsec mka session
Fri May 28 07:18:45.726 UTC

NODE: node0_0_CPU0
=============================================================================================
Interface-Name        Local-TxSCI       #Peers   Status   Key-Server   PSK/EAP      CKN
=============================================================================================
  Te0/0/0/1       6c8b.d34f.0635/0001     2      Secured      NO      FALLBACK     5556
```

Step 4 Verify whether the MKA session is secured with MACsec on the respective interface.

Example:

Before the introduction of active fallback functionality:

```
RP/0/RSP0/CPU0:router#show macsec mka session interface Fo0/0/0/1/0 detail
MKA Detailed Status for MKA Session
```

```
==================================
Status: SECURED - Secured MKA Session with MACsec
Local Tx-SCI            : 001d.e5e9.aa39/0005
Local Tx-SSCI           : 1
Interface MAC Address   : 001d.e5e9.aa39
MKA Port Identifier     : 1
Interface Name          : Fo0/0/0/1/0
CAK Name (CKN)          : 1020000000000000000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)  : A880BB45B9CE01584535F239
Message Number (MN)     : 5382
Authenticator           : NO
Key Server              : YES
MKA Cipher Suite        : AES-128-CMAC
Latest SAK Status       : Rx & Tx
Latest SAK AN           : 0
Latest SAK KI (KN)      : A880BB45B9CE01584535F23900000001 (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         : scale-21
Key Server Priority     : 20
Replay Window Size      : 40
Confidentiality Offset  : 50
Algorithm Agility       : 80C201
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 Peer List:
  MI                        MN          Rx-SCI (Peer)        SSCI KS-Priority
  -----------------------------------------------------------------------------
  4E33A276E7F79C04D80FE346       27114  d46d.5023.3704/0001     2         235
Potential Peer List:
  MI                        MN          Rx-SCI (Peer)        SSCI KS-Priority
  -----------------------------------------------------------------------------
```

! If sub-interfaces are configured, the output would be as follows:

```
RP/0/RSP0/CPU0:router# show macsec mka session interface Fo0/0/0/1/1.8 detail
MKA Detailed Status for MKA Session
==================================
Status: SECURED - Secured MKA Session with MACsec
Local Tx-SCI            : e0ac.f172.4124/001d
Local Tx-SSCI           : 1
Interface MAC Address   : e0ac.f172.4124
MKA Port Identifier     : 29
Interface Name          : Fo0/0/0/1/1.8
CAK Name (CKN)          : ABC1000000000000000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)  : 1EC4A4D1B0D75D3D5C2F6393
Message Number (MN)     : 1915
Authenticator           : NO
Key Server              : NO
MKA Cipher Suite        : AES-128-CMAC
Latest SAK Status       : Rx & Tx
Latest SAK AN           : 3
Latest SAK KI (KN)      : EB1E04894327E4EFA283C66200000003 (3)
Old SAK Status          : No Rx, No Tx
Old SAK AN              : 0
Old SAK KI (KN)         : RETIRED (4)
SAK Transmit Wait Time  : 0s (Not waiting for any peers to respond)
SAK Retire Time         : 0s (No Old SAK to retire)
```

```
MKA Policy Name         : test12
Key Server Priority     : 0
Replay Window Size      : 1024
Confidentiality Offset  : 50
Algorithm Agility       : 80C201
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 : 0
Live Peer List:
  MI                        MN          Rx-SCI (Peer)        SSCI KS-Priority
  ----------------------------------------------------------------------------
  EB1E04894327E4EFA283C662        1908  001d.e5e9.b1c0/0037     2           0
Potential Peer List:
  MI                        MN          Rx-SCI (Peer)        SSCI KS-Priority
  ----------------------------------------------------------------------------
RP/0/RSP0/CPU0:macsec-CE1#sh macsec mka  interface Fo0/0/0/1/1.8
============================================================
Interface-name          KeyChain-Name           Policy Name
============================================================
Fo0/0/0/1/1.8           kc3                     test12


! In a VPLS network with multipoint interface, the output would be as follows:

RP/0/RSP0/CPU0:router#show macsec mka session interface FortyGigE0/0/0/1/0.1 detail
MKA Detailed Status for MKA Session
===================================
Status: SECURED - Secured MKA Session with MACsec
Local Tx-SCI            : e0ac.f172.4123/0001
Local Tx-SSCI           : 1
Interface MAC Address   : e0ac.f172.4123
MKA Port Identifier     : 1
Interface Name          : Fo0/0/0/1/0.1
CAK Name (CKN)          : ABC1000000000000000000000000000000000000000000000000000000000000000000
Member Identifier (MI)  : A1DB3E42B4A543FBDBC281A6
Message Number (MN)     : 1589
Authenticator           : NO
Key Server              : NO
MKA Cipher Suite        : AES-128-CMAC
Latest SAK Status       : Rx & Tx
Latest SAK AN           : 1
Latest SAK KI (KN)      : AEC899297F5B0BDEF7C9FC6700000002 (2)
Old SAK Status          : No Rx, No Tx
Old SAK AN              : 0
Old SAK KI (KN)         : RETIRED (1)
SAK Transmit Wait Time  : 0s (Not waiting for any peers to respond)
SAK Retire Time         : 0s (No Old SAK to retire)
MKA Policy Name         : mk_xpn1
Key Server Priority     : 0
Replay Window Size      : 1024
Confidentiality Offset  : 50
Algorithm Agility       : 80C201
SAK Cipher Suite        : 0080C20001000004 (GCM-AES-XPN-256)
MACsec Capability       : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired          : YES
# of MACsec Capable Live Peers           : 2
# of MACsec Capable Live Peers Responded : 0
Live Peer List:
  MI                        MN          Rx-SCI (Peer)        SSCI KS-Priority
  ----------------------------------------------------------------------------
  AEC899297F5B0BDEF7C9FC67         225  001d.e5e9.b1bf/0001     3           0
  0A4C49EE5B7401F1BECB7E22         147  001d.e5e9.f329/0001     2           0
```

```
Potential Peer List:
  MI                        MN          Rx-SCI (Peer)        SSCI KS-Priority
  ------------------------------------------------------------------------------
```

With the introduction of active fallback functionality:

The following show command output verifies that the primary and fallback keys (CAK) are matched on both peer ends.

```
RP/0/RSP0/CPU0:router#show macsec mka session interface Hu0/0/0/11 detail

MKA Detailed Status for MKA Session
===================================
Status: Secured - Secured MKA Session with MACsec

Local Tx-SCI                    : 7061.7bea.1df4/0001
Local Tx-SSCI                   : 1
Interface MAC Address           : 7061.7bea.1df4
MKA Port Identifier             : 1
Interface Name                  : Hu0/0/0/11
CAK Name (CKN)                  : 2111
CA Authentication Mode          : PRIMARY-PSK
Keychain                        : test1
Member Identifier (MI)          : 42A78BD6243539E917B8C6B2
Message Number (MN)             : 555
Authenticator                   : NO
Key Server                      : NO
MKA Cipher Suite                : AES-128-CMAC
Configured MACSec Cipher Suite  : GCM-AES-XPN-128

Latest SAK Status               : Rx & Tx
Latest SAK AN                   : 0
Latest SAK KI (KN)              : 69B39E87B3CBA673401E989100000001 (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)
Time to SAK Rekey               : NA
Time to exit suspension         : NA

MKA Policy Name                 : P12
Key Server Priority             : 20
Delay Protection                : TRUE
Replay Window Size              : 100
Include ICV Indicator           : TRUE
Confidentiality Offset          : 0
Algorithm Agility               : 80C201
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 : 0

Live Peer List:
------------------------------------------------------------------------------
           MI                  MN            Rx-SCI          SSCI  KS-Priority
------------------------------------------------------------------------------
69B39E87B3CBA673401E9891      617       008a.96d6.194c/0001    2        20

Potential Peer List:
------------------------------------------------------------------------------
           MI                  MN            Rx-SCI          SSCI  KS-Priority
```

```
--------------------------------------------------------------------------------

Peers Status:
Last Tx MKPDU          : 2021 May 18 13:27:56.548
Peer Count             : 1

RxSCI                  : 008A96D6194C0001
  MI                    : 69B39E87B3CBA673401E9891
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 18 13:27:56.518

MKA Detailed Status for MKA Session
===================================
Status: Active - Marked Peer as Live (Waiting for SAK generation/distribution)

Local Tx-SCI                    : 7061.7bea.1df4/0001
Local Tx-SSCI                   : 1
Interface MAC Address           : 7061.7bea.1df4
MKA Port Identifier             : 1
Interface Name                  : Hu0/0/0/11
CAK Name (CKN)                  : 2000
CA Authentication Mode          : FALLBACK-PSK
Keychain                        : test1f
Member Identifier (MI)          : 1BB9428C721F6EE3E538C942
Message Number (MN)             : 553
Authenticator                   : NO
Key Server                      : NO
MKA Cipher Suite                : AES-128-CMAC
Configured MACSec Cipher Suite  : GCM-AES-XPN-128

Latest SAK Status               : Rx & Tx
Latest SAK AN                   : 0
Latest SAK KI (KN)              : 69B39E87B3CBA673401E989100000001 (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)
Time to SAK Rekey               : NA
Time to exit suspension         : NA

MKA Policy Name                 : P12
Key Server Priority             : 20
Delay Protection                : TRUE
Replay Window Size              : 100
Include ICV Indicator           : TRUE
Confidentiality Offset          : 0
Algorithm Agility               : 80C201
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 : 0

Live Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------
8F59AD6021FA3E2D5F9E6231      615       008a.96d6.194c/0001    2        20

Potential Peer List:
--------------------------------------------------------------------------------
```

```
           MI                  MN             Rx-SCI          SSCI  KS-Priority
-----------------------------------------------------------------------------

Peers Status:
Last Tx MKPDU          : 2021 May 18 13:27:56.547
Peer Count             : 1

RxSCI                  : 008A96D6194C0001
  MI                   : 8F59AD6021FA3E2D5F9E6231
  Peer CAK             : Match
  Latest Rx MKPDU      : 2021 May 18 13:27:56.518

RP/0/RSP0/CPU0:router#
```

If sub-interfaces are configured, the output would be as follows. In this example, the status of FALLBACK-PSK is *Secured*.

```
RP/0/RSP0/CPU0:router# show macsec mka session interface Hu0/0/0/0.6 detail
MKA Detailed Status for MKA Session
==================================
Status: Secured - Secured MKA Session with MACsec

Local Tx-SCI                    : 7061.7bea.1dc8/0006
Local Tx-SSCI                   : 1
Interface MAC Address           : 7061.7bea.1dc8
MKA Port Identifier             : 6
Interface Name                  : Hu0/0/0/0.6
CAK Name (CKN)                  : 9999
CA Authentication Mode          : FALLBACK-PSK
Keychain                        : D_tagf
Member Identifier (MI)          : 1DE18714A098B80964CC651E
Message Number (MN)             : 6203
Authenticator                   : NO
Key Server                      : YES
MKA Cipher Suite                : AES-128-CMAC
Configured MACSec Cipher Suite  : GCM-AES-XPN-256

Latest SAK Status               : Rx & Tx
Latest SAK AN                   : 0
Latest SAK KI (KN)              : 1DE18714A098B80964CC651E00000001 (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)
Time to SAK Rekey               : 23510s
Time to exit suspension         : NA

MKA Policy Name                 : D_tag1
Key Server Priority             : 1
Delay Protection                : FALSE
Replay Window Size              : 1000
Include ICV Indicator           : TRUE
Confidentiality Offset          : 50
Algorithm Agility               : 80C201
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
```

```
# of MACSec Suspended Peers              : 0

Live Peer List:
----------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
----------------------------------------------------------------------------
5C852D8F920306893D2BFB8F     10978      00c1.645f.2dd4/0006    2        11

Potential Peer List:
----------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
----------------------------------------------------------------------------

Suspended Peer List:
----------------------------------------------------------------------------
       Rx-SCI          SSCI
----------------------------------------------------------------------------

Peers Status:
Last Tx MKPDU          : 2021 May 18 13:29:15.687
Peer Count             : 1

RxSCI                  : 00C1645F2DD40006
  MI                    : 5C852D8F920306893D2BFB8F
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 18 13:29:15.769

RP/0/RSP0/CPU0:router#


! In a VPLS network with multipoint interface, the output would be as follows:

RP/0/RSP0/CPU0:router#show macsec mka session interface Hu0/0/1/7 detail
Fri May 28 07:19:11.362 UTC

MKA Detailed Status for MKA Session
==================================
Status: Secured - Secured MKA Session with MACsec

Local Tx-SCI                   : 6c8b.d34f.0635/0001
Local Tx-SSCI                  : 2
Interface MAC Address          : 6c8b.d34f.0635
MKA Port Identifier            : 1
Interface Name                 : Te0/0/0/1
CAK Name (CKN)                 : 5556
CA Authentication Mode         : FALLBACK-PSK
Keychain                       : test2f
Member Identifier (MI)         : 6D14ECCDFB70E7E0463BD509
Message Number (MN)            : 20455
Authenticator                  : NO
Key Server                     : NO
MKA Cipher Suite               : AES-256-CMAC
Configured MACSec Cipher Suite : GCM-AES-XPN-256

Latest SAK Status              : Rx & Tx
Latest SAK AN                  : 2
Latest SAK KI (KN)             : 1BBDDC0520C797C26AB7F1BF00000002 (2)
Old SAK Status                 : No Rx, No Tx
Old SAK AN                     : 1
Old SAK KI (KN)                : RETIRED (1)

SAK Transmit Wait Time         : 0s (Not waiting for any peers to respond)
```

```
SAK Retire Time                 : 0s (No Old SAK to retire)
Time to SAK Rekey               : NA
Time to exit suspension         : NA

MKA Policy Name                 : *DEFAULT POLICY*
Key Server Priority             : 16
Delay Protection                : FALSE
Replay Window Size              : 64
Include ICV Indicator           : FALSE
Confidentiality Offset          : 0
Algorithm Agility               : 80C201
SAK Cipher Suite                : 0080C20001000004 (GCM-AES-XPN-256)
MACsec Capability               : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired                  : YES

# of MACsec Capable Live Peers           : 2
# of MACsec Capable Live Peers Responded : 0

Live Peer List:
------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
------------------------------------------------------------------------------
1BBDDC0520C797C26AB7F1BF     19997      008a.96d6.194c/0001    3        16
B25B1000CC6FAE92D1F85738      139       dc77.4c3e.59c3/0001    1        16

Potential Peer List:
------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
------------------------------------------------------------------------------

Peers Status:
 Last Tx MKPDU          : 2021 May 28 07:19:10.153
 Peer Count             : 2

 RxSCI                  : 008A96D6194C0001
  MI                    : 1BBDDC0520C797C26AB7F1BF
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 28 07:19:09.960

 RxSCI                  : DC774C3E59C30001
  MI                    : B25B1000CC6FAE92D1F85738
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 28 07:19:10.180

RP/0/RSP0/CPU0:router#


RP/0/RSP0/CPU0:router#show macsec mka session interface Hu0/0/1/7.1 detail

MKA Detailed Status for MKA Session
===================================
Status: Secured - Secured MKA Session with MACsec

Local Tx-SCI                    : 7061.7bff.e5e8/0001
Local Tx-SSCI                   : 2
Interface MAC Address           : 7061.7bff.e5e8
MKA Port Identifier             : 1
Interface Name                  : Hu0/0/1/7.1
CAK Name (CKN)                  : 5556
CA Authentication Mode          : FALLBACK-PSK
Keychain                        : test22f
Member Identifier (MI)          : 8FF3D1BBF09EA4AD6A0FC1B5
Message Number (MN)             : 81
Authenticator                   : NO
```

```
Key Server                      : YES
MKA Cipher Suite                : AES-256-CMAC
Configured MACSec Cipher Suite  : GCM-AES-XPN-256

Latest SAK Status               : Rx & Tx
Latest SAK AN                   : 3
Latest SAK KI (KN)              : 8FF3D1BBF09EA4AD6A0FC1B500000002 (2)
Old SAK Status                  : No Rx, No Tx
Old SAK AN                      : 2
Old SAK KI (KN)                 : RETIRED (1)

SAK Transmit Wait Time          : 0s (Not waiting for any peers to respond)
SAK Retire Time                 : 0s (No Old SAK to retire)
Time to SAK Rekey               : 17930s
Time to exit suspension         : NA

MKA Policy Name                 : P123
Key Server Priority             : 10
Delay Protection                : FALSE
Replay Window Size              : 64
Include ICV Indicator           : FALSE
Confidentiality Offset          : 30
Algorithm Agility               : 80C201
SAK Cipher Suite                : 0080C20001000004 (GCM-AES-XPN-256)
MACsec Capability               : 3 (MACsec Integrity, Confidentiality, & Offset)
MACsec Desired                  : YES

# of MACsec Capable Live Peers           : 2
# of MACsec Capable Live Peers Responded : 2

# of MACSec Suspended Peers              : 0

Live Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------
6BCF91135F807CB9F57DDAAA       61       dc77.4c3e.5b05/0001    1        24
D81CFE93D07E932DDC33666E       44       00a7.4250.56c2/0001    3        25

Potential Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------

Suspended Peer List:
--------------------------------------------------------------------------------
       Rx-SCI          SSCI
--------------------------------------------------------------------------------

Peers Status:
 Last Tx MKPDU          : 2021 May 28 13:16:50.992
 Peer Count             : 2

 RxSCI                  : DC774C3E5B050001
  MI                    : 6BCF91135F807CB9F57DDAAA
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 28 13:16:51.312

 RxSCI                  : 00A7425056C20001
  MI                    : D81CFE93D07E932DDC33666E
  Peer CAK              : Match
  Latest Rx MKPDU       : 2021 May 28 13:16:50.945
RP/0/RSP0/CPU0:router#
```

Step 5 Verify the MACsec session counter statistics.

Example:

```
RP/0/RSP0/CPU0:router# show macsec mka statistics interface Fo0/0/0/1/0


MKA Statistics for Session on interface (Fo0/0/0/1/0)
=====================================================
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.......... 3
SAKs Rekeyed............ 2
SAKs Received........... 0
SAK Responses Received.. 3

MKPDU Statistics
MKPDUs Transmitted...... 5425
"Distributed SAK".. 8
"Distributed CAK".. 0
MKPDUs Validated & Rx... 4932
"Distributed SAK".. 0
"Distributed CAK".. 0

MKA IDB Statistics
MKPDUs Tx Success.......... 5425
MKPDUs Tx Fail............. 0
MKPDUS Tx Pkt build fail... 0
MKPDUs Rx CA Not found..... 0
MKPDUs Rx Error............ 0
MKPDUs Rx Success.......... 4932

MKPDU Failures
   MKPDU Rx Validation (ICV)............... 0
   MKPDU Rx Bad Peer MN.................... 0
   MKPDU Rx Non-recent Peerlist MN........ 0
   MKPDU Rx Drop SAKUSE, KN mismatch...... 0
   MKPDU Rx Drop SAKUSE, Rx Not Set....... 0
   MKPDU Rx Drop SAKUSE, Key MI mismatch.. 0
   MKPDU Rx Drop SAKUSE, AN Not in Use.... 0
   MKPDU Rx Drop SAKUSE, KS Rx/Tx Not Set. 0

SAK Failures
   SAK Generation................... 0
   Hash Key Generation.............. 0
   SAK Encryption/Wrap.............. 0
   SAK Decryption/Unwrap............ 0

! If sub-interfaces are configured, the output would be as follows:

RP/0/RSP0/CPU0:router# show macsec mka statistics interface Fo0/0/0/1/1.8

MKA Statistics for Session on interface (Fo0/0/0/1/1.8)
=======================================================
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.......... 0
   SAKs Rekeyed............ 0
   SAKs Received........... 9
   SAK Responses Received.. 0
MKPDU Statistics
   MKPDUs Transmitted...... 1973
      "Distributed SAK".. 0
      "Distributed CAK".. 0
   MKPDUs Validated & Rx... 1965
      "Distributed SAK".. 9
      "Distributed CAK".. 0
MKA IDB Statistics
   MKPDUs Tx Success.......... 1973
   MKPDUs Tx Fail............. 0
   MKPDUS Tx Pkt build fail... 0
   MKPDUs Rx CA Not found..... 0
   MKPDUs Rx Error............ 0
   MKPDUs Rx Success.......... 1965

! In a VPLS network with a mulitpoint interface, the output would be as follows:

RP/0/RSP0/CPU0:router# show macsec mka statistics interface FortyGigE0/0/0/1/0.1

MKA Statistics for Session on interface (Fo0/0/0/1/0.1)
=======================================================
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.......... 0
   SAKs Rekeyed............ 0
   SAKs Received........... 2
   SAK Responses Received.. 0
MKPDU Statistics
   MKPDUs Transmitted...... 1608
      "Distributed SAK".. 0
      "Distributed CAK".. 0
   MKPDUs Validated & Rx... 406
      "Distributed SAK".. 2
      "Distributed CAK".. 0
MKA IDB Statistics
   MKPDUs Tx Success.......... 1608
   MKPDUs Tx Fail............. 0
   MKPDUS Tx Pkt build fail... 0
   MKPDUs Rx CA Not found..... 0
   MKPDUs Rx Error............ 0
   MKPDUs Rx Success.......... 1802
```

The counters display the MACsec PDUs transmitted, validated, and received. The output also displays transmission errors, if any.

This completes the verification of MACsec encryption on the IOS-XR.

Verifying MACsec Encryption on ASR 9000

MACsec encryption on the router hardware can be verified by running relevant commands in the Privileged Executive Mode.

To verify if MACsec encryption has been correctly configured, follow these steps.

SUMMARY STEPS

1. Verify the MACsec encryption and hardware interface descriptor block (IDB) information on the interface.
2. Use the IDB handle retrieved from Step 1 to verify the platform hardware information.
3. Use the Transmitter SA retrieved from Step 2 to verify the MACsec SA information programmed in the hardware.
4. Verify the MACsec Secure Channel (SC) information programmed in the hardware.

DETAILED STEPS

Procedure

Step 1 Verify the MACsec encryption and hardware interface descriptor block (IDB) information on the interface.

Example:

```
RP/0/RSP0/CPU0:router# show macsec ea idb interface Fo0/0/0/1/0


IDB Details:
if_sname : Fo0/0/0/1/0
if_handle : 0x3480
Replay window size : 64
Local MAC : 00:1d:e5:e9:aa:39
Rx SC Option(s) : Validate-Frames Replay-Protect
Tx SC Option(s) : Protect-Frames Always-Include-SCI
Security Policy : MUST SECURE
Sectag offset : 8
VLAN : Outer tag (etype=0x8100, id=1, priority=0, cfi=0): Inner tag (etype=0x8100, id=1, priority=0,
 cfi=0)
Rx SC 1
Rx SCI : 001de5e9b1bf0019
Peer MAC : 00:1d:e5:e9:b1:bf
Stale : NO
SAK Data
SAK[0] : ***
SAK Len : 32
HashKey[0] : ***
HashKey Len : 16
Conf offset : 30
Cipher Suite : GCM-AES-XPN-256
CtxSalt[0] : 83 c3 7b ad 7b 6f 63 16 09 8f f3 d2
Rx SA Program Req[0]: 2015 Oct 09 15:20:53.082
Rx SA Program Rsp[0]: 2015 Oct 09 15:20:53.092

Tx SC
Tx SCI : 001de5e9aa39001a
Active AN : 0
Old AN : 255
Next PN : 1, 0, 0, 0
```

```
SAK Data
SAK[0] : ***
SAK Len : 32
HashKey[0] : ***
HashKey Len : 16
Conf offset : 30
Cipher Suite : GCM-AES-XPN-256
CtxSalt[0] : 83 c3 7b ae 7b 6f 63 16 09 8f f3 d2
Tx SA Program Req[0]: 2015 Oct 09 15:20:55.053
Tx SA Program Rsp[0]: 2015 Oct 09 15:20:55.064
```

! When more than 1 RX SA is configured in P2MP networks, the output would be as follows:

```
RP/0/RSP0/CPU0:router# show macsec ea idb interface FortyGigE0/0/0/1/0.1
IDB Details:
  if_sname                : Fo0/0/0/1/0.1
  if_handle               : 0x2e40
  Replay window size      : 1024
  Local MAC               : e0:ac:f1:72:41:23
  Rx SC Option(s)         : Validate-Frames Replay-Protect
  Tx SC Option(s)         : Protect-Frames Always-Include-SCI
  Security Policy         : MUST SECURE
  Sectag offset           : 8
  VLAN                    : Outer tag (etype=0x8100, id=1, priority=0, cfi=0)
                          : Inner tag (etype=0x8100, id=1, priority=0, cfi=0)
  Rx SC 1
    Rx SCI                : 001de5e9f3290001
    Peer MAC              : 00:1d:e5:e9:f3:29
    Stale                 : NO
    SAK Data
      SAK[1]              : ***

      SAK Len             : 32
      HashKey[1]          : ***
      HashKey Len         : 16
      Conf offset         : 50
      Cipher Suite        : GCM-AES-XPN-256
      CtxSalt[1]          : ae ca 99 2b 7f 5b 0b de f7 c9 fc 67
  Rx SC 2
    Rx SCI                : 001de5e9b1bf0001
    Peer MAC              : 00:1d:e5:e9:b1:bf
    Stale                 : NO
    SAK Data
      SAK[1]              : ***

      SAK Len             : 32
      HashKey[1]          : ***
      HashKey Len         : 16
      Conf offset         : 50
      Cipher Suite        : GCM-AES-XPN-256
      CtxSalt[1]          : ae ca 99 2a 7f 5b 0b de f7 c9 fc 67
  Tx SC
    Tx SCI                : e0acf17241230001
    Active AN             : 1
    Old AN                : 0
    Next PN               : 1, 1, 0, 0
    SAK Data
      SAK[1]              : ***

      SAK Len             : 32
      HashKey[1]          : ***
      HashKey Len         : 16
      Conf offset         : 50
```

```
Cipher Suite        : GCM-AES-XPN-256
CtxSalt[1]          : ae ca 99 28 7f 5b 0b de f7 c9 fc 67
```

The **if_handle** field provides the IDB instance location.

The **Replay window size** field displays the configured window size.

The **Security Policy** field displays the configured security policy.

The **Local Mac** field displays the MAC address of the router.

The **Peer Mac** field displays the MAC address of the peer. This confirms that a peer relationship has been formed between the two routers.

Step 2 Use the IDB handle retrieved from Step 1 to verify the platform hardware information.

Example:

```
RP/0/RSP0/CPU0:router# show macsec ea platform hardware
idb location 0/0/CPU0 | b 3480

if_handle : 0x00003480
NPPort : 099 [0x063]
LdaPort : 016 [0x010] SerdesPort : 000 [0x000]
NetSoftPort : 061 [0x03d] SysSoftPort : 062 [0x03e]
Active AN : 0x00000000 Idle AN : 0x000000ff
Match-All Tx SA : 0x80010001 Match-All Rx SA : 0x00010001
Match-All Tx Flow : 0x80000003 Match-All Rx Flow : 0x00000003
Bypass Tx SA : 0x80000000 Bypass Rx SA : 0x00000000
Tx SA[0] : 0x80020002 Tx Flow[0] : 0x8000000c
Tx SA[1] : 0xffffffff Tx Flow[1] : 0xffffffff
Tx SA[2] : 0xffffffff Tx Flow[2] : 0xffffffff
Tx SA[3] : 0xffffffff Tx Flow[3] : 0xffffffff
Rx SA[0] : 0x00020002 Rx Flow[0] : 0x0000000c
Rx SA[1] : 0xffffffff Rx Flow[1] : 0xffffffff
Rx SA[2] : 0xffffffff Rx Flow[2] : 0xffffffff
Rx SA[3] : 0xffffffff Rx Flow[3] : 0xffffffff
```

Step 3 Use the Transmitter SA retrieved from Step 2 to verify the MACsec SA information programmed in the hardware.

Example:

```
RP/0/RSP0/CPU0:router# show macsec ea platform hardware sa
0x80020002 interface Fo0/0/0/1/0 location 0/0/CPU0

MACSEC HW SA Details:
Action Type : 0x00000003
Direction : Egress
Dest Port : 0x00000000
Conf Offset : 00000030
Drop Type : 0x00000002
Drop NonResvd : 0x00000000
SA In Use : YES
ConfProtect : YES
IncludeSCI : YES
ProtectFrame : YES
UseEs : NO
UseSCB : NO
SCI : 00 1d e5 e9 aa 39 00 05
Replay Window : 64 MacsecCryptoAlgo : 7
Direction : Egress AN : 0
```

```
AES Key Len : 256 X-Packet Number : 0x0000000000000000
CtxSalt : f8d88dc3e1c5e6a94ca2299
```

The output displays the details of the encryption, such as the AES key, the Auth key, and other parameters.

Step 4 Verify the MACsec Secure Channel (SC) information programmed in the hardware.

Example:

```
RP/0/RSP0/CPU0:router# show macsec ea platform hardware msc
interface Fo0/0/0/1/0 location 0/0/CPU0


MACSEC HW Cfg Details:
Mode : 0x5
Counter Clear on Read : 0x0
SA Fail Mask : 0xffff
VlanCounter Update : 0x1
Global SecFail Mask : 0xffffffff
Latency : 0xff
StaticBypass : 0x0
Should secure : 0x0
Global Frame Validation : 0x2
Ctrl Pkt CC Bypass : 0x1
NonCtrl Pkt CC Bypass : 0x1
Sequence Number Threshold : 0xbffffb8
Sequence Number Threshold 64bit : 0x000002fffffffffd
Non Matching Non Control Pkts Programming
      Untagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
      Tagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
      BadTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
      KayTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
Non Matching Control Pkts Programming
      Untagged : Bypass: 0x1 DestPort : 0x2, DropType : 0xffffffff
      Tagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
      BadTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
      KayTagged : Bypass: 0x0 DestPort : 0x2, DropType : 0x2
```

This completes the verification of MACsec encryption on the router hardware.

This completes the configuration and verification of MACsec encryption.

Quantum safe key distribution options for MACsec

Quantum computers are a threat to existing cryptographic algorithms. To address this problem, you can use session keys to establish a secure connection between two routers.

Cisco offers two solutions to derive session keys:

- Session Key Service (SKS): Used to derive the session keys on both the routers establishing the MACsec connection without using an external key source.
- Secure Key Integration Protocol (SKIP): Used to derive the session keys on both the routers establishing the MACsec connection using an external server. Enables a router to securely import a post-quantum pre-shared key (PPK) from an external key source such as a quantum key distribution (QKD) device.

Table 35: Feature History Table

Feature Name	Release Information	Feature Description
Session key service	Release 7.9.1	The router integrates the Session Key Service (SKS) as a software component, allowing it to generate and manage the cryptographic keys needed for quantum-safe MACsec. By using SKS, you can implement MACsec without requiring additional hardware, simplifying deployment and reducing costs. The SKS software should be present on the peer routers. For more information on Quantum Key Distribution, see Post Quantum Security Brief.

Session key service

The Session Key Service (SKS) is a cryptographic service that manages symmetric keys for encryption and decryption. These are the salient features of SKS on a router.

Key generation

The SKS engine is a software component within the router responsible for generating cryptographic keys. It creates keys that will be used to encrypt and decrypt data between peer routers.

No additional hardware required

This implies that the key generation and exchange process does not need extra hardware components. The SKS engine functions with the existing router hardware, making it cost-effective and easy to deploy.

Seed protected by McEliece cryptosystem

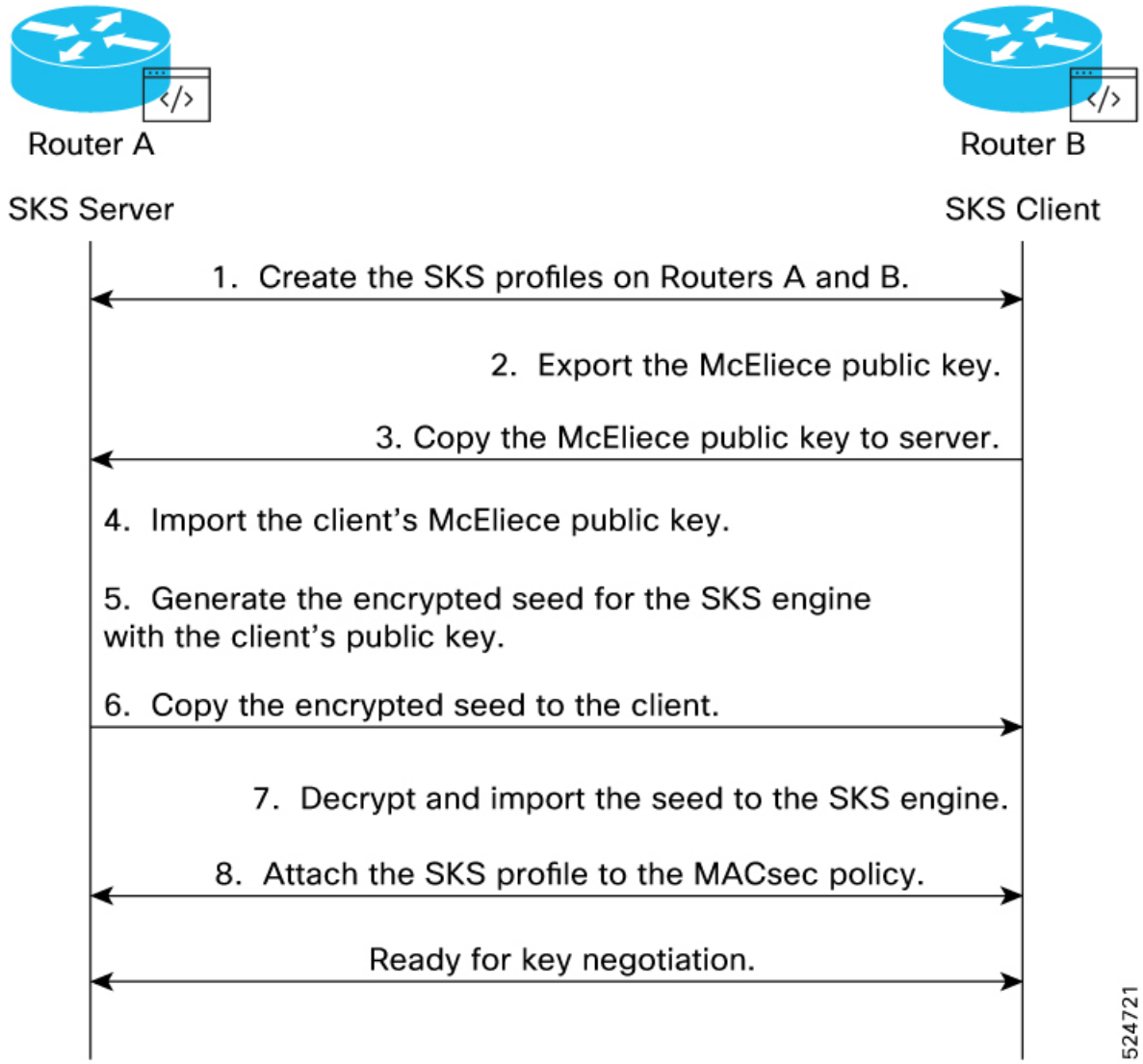
Seeding refers to initializing the SKS with a specific value, called a seed, which ensures that both communicating peers generate the same cryptographic keys. This is crucial for successful encryption and decryption. The McEliece cryptosystem is a public-key cryptosystem known for its resistance to quantum computer attacks. Protecting the seed with McEliece ensures it remains secure against future quantum computing threats.

Only Key ID sent on the network

Instead of sending the entire cryptographic key over the network, the router sends only a Key Identifier (Key ID). The receiving peer uses the Key ID to derive the corresponding key locally using its SKS. Using the Key ID enhances security by minimizing exposure of the actual key during transmission.

Configure SKS

The SKS software component on a router is used to configure the server and client to generate preshared keys for quantum-safe MACsec. This image depicts the steps you should perform to generate preshared keys. You can click a step to learn more.



1 Create the SKS profiles on Routers A and B.
2 Export the McEliece public key.
3 Copy the McEliece public key to the server.
4 Import the client's McEliece public key.
5 Generate the encrypted seed for the SKS engine with the clients public key.
6 Copy the encrypted seed to the client.
7 Decrypt and import the seed to the sks engine.
8 Attach the SKS profile to the MACsec policy.
9 Ready for key negotiation.

Create the SKS profile on the server and client

Follow these steps to create the SKS profile on the server and client.

Before you begin

First, let us gather the required details to facilitate the devices to agree on encryption parameters to establish a secure connection:

- Routers A and B are equipped with the SKS engine.
- Router A acts as the server and Router B acts as the client.
- Routers A and B are ready for key negotiation after all the steps are performed and the SKS profile is attached to the MACsec policy.

Procedure

Step 1 Enter the **sks profile** <*profile-name*> **device-identifier** <*name of peer*> command to apply a profile to manage secure communications.

Server configuration

```
Router A(config)# sks profile prof-A device-identifier peer-1
```

Client configuration

```
Router B(config)#sks profile prof-B device-identifier peer-2
```

Step 2 Enter the **live-key** <*number of MACsec sessions*> command to manage the active keys used in MACsec sessions.

Server configuration

```
Router A(config-sks-profile)#live-keys 5
```

Client configuration

```
Router B(config-sks-profile)#live-keys 5
```

Step 3 Enter the **peer identifier** <*peer-name*> command to associate the server with the client and the client with the server.

Server configuration

```
Router A(config-sks-profile)#peer-identifier peer-2
```

Client configuration

```
Router B(config-sks-profile)#peer-identifier peer-1 master
```

Export the McEliece public key

Follow these steps to export the McEliece public key.

Procedure

Step 1 Enter the **crypto sks key export mceliece** command to export the public key to the client.

Client configuration

```
Router B#crypto sks key export mceliece
```

The message **Pubkey exported file: disk0:/MeCe_the_MC_default_pub** is displayed.

Step 2 Verify the default path where the public key is exported.

Client configuration

```
Router B#dir disk0:/MeCe_the_MC_default_pub
Mon Feb 24 04:25:25.013 UTC

Directory of disk0:/MeCe_the_MC_default_pub
73 -rw-r--r--. 1 1357824 Feb 24 04:20 MeCe_the_MC_default_pub

989244 kbytes total (919872 kbytes free)
```

Copy the McEliece public key to the server

Follow these steps to copy the McEliece public key to the server.

Before you begin

In the example, **disk0:/MeCe_the_MC_default_pub** is the source path of the client and **cisco@1.2.42.3** is the IP address of the server.

Procedure

Step 1 Using the Secure Copy Protocol (SCP), enter the **scp / disk0\:/**<*source path on the client* <*destination path of the server*>**:/disk0:/** to copy the files from the client to the server.

Client configuration

```
Router B# scp /disk0\:/MeCe_the_MC_default_pub cisco@1.2.42.3:/disk0:/
```

This command has securely copied the file named **MeCe_the_MC_default_pub** from the local directory **/disk0:/** to the remote directory **/disk0:/** on the host **1.2.42.3** using the Cisco user account.

Step 2 Verify that the files are copied from the client to the server.

Client configuration

```
Router B #dir disk0:/MeCe_the_MC_default_pub
Mon Feb 24 04:27:00.398 UTC

Directory of disk0:/MeCe_the_MC_default_pub
73 -rw-r--r--. 1 1357824 Feb 24 04:26 MeCe_the_MC_default_pub

989244 kbytes total (919868 kbytes free)
```

Import the client McEliece public key

Follow these steps to import the client McEliece public key.

Before you begin

In the example, peer 2 is Router B's name to which the key corresponds. The local directory **disk0:/MeCe_the_MC_default_pub** is the source path of the key on the server that was copied in the previous step.

Procedure

Step 1 Enter the **crypto sks key import mceliece** *<client's name> <source path of the key on the server>* command to import the McEliece public key to the server.

Server configuration

```
Router A# crypto sks key import mceliece peer-2 disk0:/MeCe_the_MC_default_pub
```

Step 2 Verify that the **Pubkey Import Done** is set to **True** for the required peer.

Server configuration

```
Router A# show crypto sks peer all
Mon Feb 24 04:29:06.492 UTC
Peer Name          : peer-2
Profile Name       : prof-A
Seed Done          : TRUE
Pubkey Import Done : TRUE
Master             : FALSE
```

Generate the encrypted seed for the SKS engine with the client public key

Follow these steps to generate the encrypted seed for the SKS engine with the client public key.

Procedure

Step 1 Enter the **crypto sks seed export mceliece** *<client name>* command to generate and export the seed to the server.

Server configuration

```
Router A# crypto sks seed export mceliece peer-2
```

The command exports a McEliece cryptographic seed that is associated with Router B.

Step 2 Verify if an encrypted seed is exported to the */disk0\:/enc_self_peer-2* location.

Server configuration

```
Router A# dir disk0:/enc_self_peer-2
Mon Feb 24 04:35:57.596 UTC

Directory of disk0:/enc_self_peer-2
74 -rw-r--r--. 1 480 Feb 24 04:35 enc_self_peer-2

989244 kbytes total (919860 kbytes free)
```

Copy the encrypted seed to the client

Follow these steps to copy the encrypted seed to the client.

Procedure

Step 1 Using the Secure Copy Protocol (SCP), enter the **scp /disk0\:/**<*source path of the client*> <*destination path of the server*> command to copy the files from the server to the client.

Server configuration

```
Router A# scp /disk0\:/enc_self_peer-2 cisco@1.2.43.3:/disk0:/
```

Step 2 Verify that the files are copied from the server to the client.

Client configuration

```
Router B# dir disk0:/enc_self_peer-2
Mon Feb 24 04:35:57.596 UTC

Directory of disk0:/enc_self_peer-2
74 -rw-r--r--. 1 480 Feb 24 04:35 enc_self_peer-2

989244 kbytes total (919860 kbytes free)
```

Decrypt and import the seed to the sks engine

Follow these steps to decrypt and import the seed to the sks engine.

Procedure

Step 1 Enter the **crypto sks seed import mceliece** <*server name*> <*server path*> command to import the seed on the client.

Client configuration

```
Router B# crypto sks seed import mceliece peer-1 disk0:/enc_self_peer-2
```

The seed is associated with Router A and **disk0:/enc_self_peer-2** is the file path from which the seed is being imported. It indicates that the seed is stored in a file located at **disk0:/enc_self_peer-2** on the router.

Step 2 Verify that the seed is imported to the client.

Client configuration

```
Router B# show crypto sks peer all
Mon Feb 24 04:37:35.578 UTC
Peer Name          : peer-1
Profile Name       : prof-B
Seed Done          : TRUE
Pubkey Import Done : FALSE
Master             : TRUE
------------------------------
```

Attach the SKS profile to the MACsec policy

Follow these steps to create the SKS profile on the server and client.

Procedure

Step 1 Configure the SKS profile on both MACsec peers.

Server configuration

```
Router A(config)#macsec-policy p1
Router A(config-macsec-policy-p1)#ppk
Router A(config-macsec-policy-p1-ppk)#sks-profile prof-A
```

Client configuration

```
Router B(config)#macsec-policy p2
Router B(config-macsec-policy-p1)#ppk
Router B(config-macsec-policy-p1-ppk)#sks-profile prof-B
```

Step 2 Verify the MACsec configuration.

Server configuration

```
Router A#show macsec mka session

MKA Detailed Status for MKA Session
===================================
Status: Secured - Secured MKA Session with MACsec

Local Tx-SCI                    : c847.091c.d060/0001
Local Tx-SSCI                   : 1
Interface MAC Address           : c847.091c.d060
MKA Port Identifier             : 1
Interface Name                  : Te0/0/0/24
CAK Name (CKN)                  : 4000
CA Authentication Mode          : PRIMARY-PSK
Keychain                        : tet
Member Identifier (MI)          : 2E222A17F6E9535E1ACD4747
Message Number (MN)             : 333807
Authenticator                   : NO
Key Server                      : NO
MKA Cipher Suite                : AES-256-CMAC
Configured MACSec Cipher Suite  : GCM-AES-XPN-256
Key Distribution Mode           : PPK
------------<truncated>--------------
```

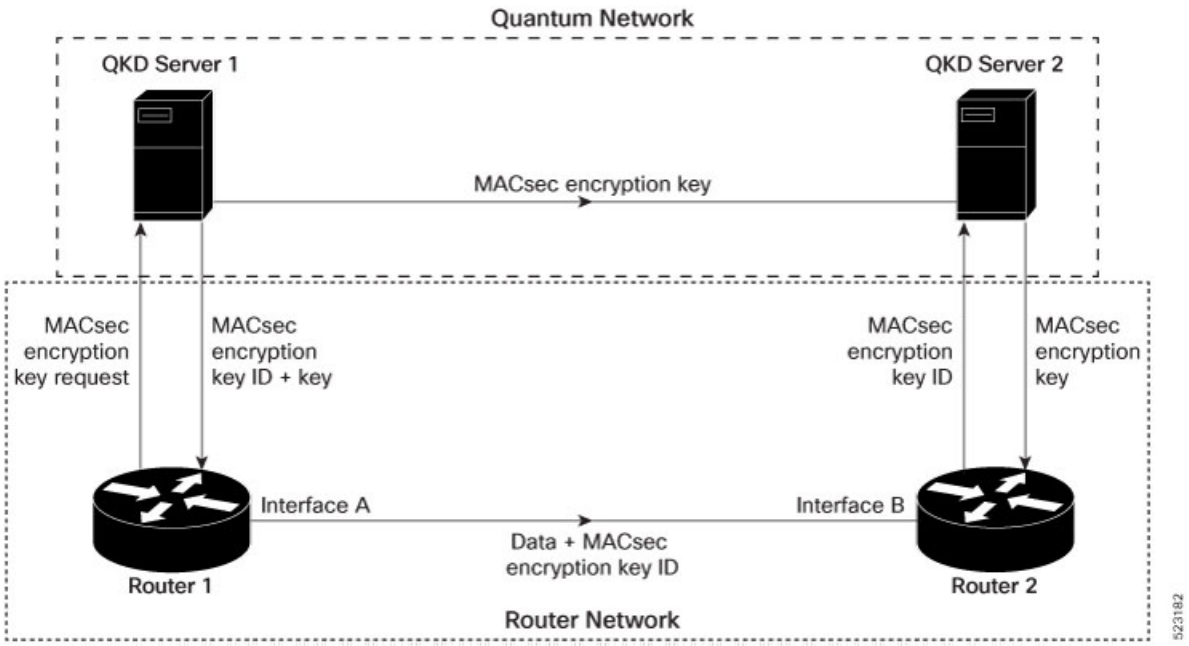
Ready for key negotiation: Once the routers attach the SKS profile to the MACsec policy, they have met all technical and security prerequisites. Routers A and B can now establish a secure communication link using symmetric key cryptography.

Understanding SKIP

Table 36: Feature History Table

Feature Name	Release Information	Feature Description
Secure Key Integration Protocol for Cisco IOS XR Routers	Release 7.10.1	We have now enabled Secure Key Integration Protocol (SKIP), a key-exchange protocol, on your routers to ensure a long-term secure MACsec. This is made possible because the SKIP protocol facilitates communication with external quantum devices, thereby enabling your routers to use Quantum Key Distribution (QKD) to create and transmit secure MACsec keys. Using QKD overcomes a critical problem in a post-quantum world where the current cryptographic systems are no longer secure due to the advent of quantum computers. This feature introduces these changes: • **CLI:** • **crypto-sks-kme** • **show crypto sks profile** • **Yang Data Model**: Cisco-IOS-XR-um-sks-server-cfg.yang (see GitHub, YANG Data Models Navigator)

Cisco Secure Key Integration Protocol (SKIP) enables your router that supports encryption to use keys by a quantum distribution system. SKIP implementation in Cisco IOS XR software supports integrating external Quantum Key Distribution (QKD) devices with your routers. With integration support between the routers and QKD devices, you can use the QKD devices to exchange encryption keys for communication between the routers. And this mechanism eliminates the key distribution problem in a post quantum world.



Quantum Key Distribution (QKD) is a method for securely transmitting a secret key between two parties. QKD uses the laws of quantum mechanics to guarantee security even when eavesdroppers monitor the communication channel. In QKD, the key is encoded in the states of single photons. The QKD transmits the keys over optical fiber or free space (vacuum). The security of the key relies on the fact that measuring a quantum state introduces a change in the quantum state. The change in quantum states helps the two end parties of the communication channel to identify any interception of their key.

QKD is a secure key exchange mechanism against quantum attacks and will remain so, even with future advancements in cryptanalysis or quantum computing. Unlike other cryptographic algorithms, QKD doesn't need continual updates based on discovered vulnerabilities.

Feature Highlights

- You can use the QKD devices in the following combinations:
  - Same QKD device on the end ports of the peer routers
  - Different QKD devices on the end ports of the peer routers
  - Multiple links between the same peer routers using different QKD devices
- You can use a specific source interface for the router communication with the QKD devices. To use a specific source interface, configure the source interface in the QKD profile. Use the **source interface** command in SKS configuration mode as follows.

```
Router# config
Router(config)# sks profile ProfileR1toR2 type remote
Router(config-sks-profile)# kme server ipv4 192.0.2.34 port 10001
Router(config-sks-profile)# source interface hundredGigE 0/1/0/17
Router(config-sks-profile)# commit
```

- You can use an HTTP Proxy for the router communication with the QKD devices. Use the following configuration for the router to use an HTTP proxy server to communicate to the QKD devices.

```
Router# config
Router(config)# sks profile ProfileR1toR2 type remote
Router(config-sks-profile)# kme server ipv4 192.0.2.34 port 10001
Router(config-sks-profile)# http proxy ipv4 192.0.2.68 port 804
Router(config-sks-profile)# commit
```

Note The **http proxy server** command supports configuration using IPv4 address, IPv6 address, and hostname of the HTTP proxy.

Restrictions

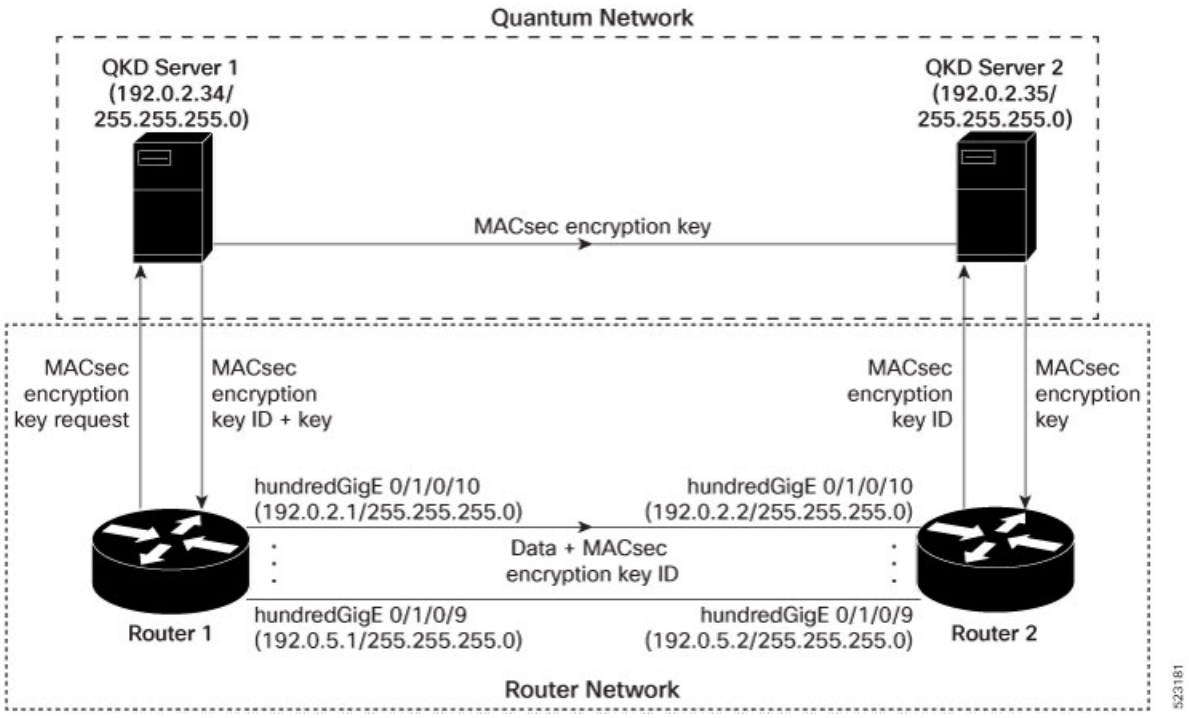
The following section lists the restriction to consider prior implementing SKIP:

- You can use the SKIP protocol only in a Point to Point MACsec link encryption scenario.
- The SKIP protocol is available only on the interfaces that support MACsec encryption.

Configure Point-to-Point MACsec Link Encryption using SKIP

In Point-to-Point MACsec Link Encryption, the router uses SKIP to establish secure encryption. This encryption is set up between two interfaces in peer routers and requires the assistance of an external QKD device network. The QKD network shares the MACsec encryption key instead of the router network. Thus, when the router needs to create a MACsec link between peer router interfaces, it contacts the external QKD device and requests the key. The external QKD device generates a Key pair comprising the Key ID and the Key. The Key ID serves as the unique identification string for the Key (Shared Secret). The QKD then shares both the Key ID and Key with the router and the router shares only the Key ID with its peer. The Peer router uses this Key ID to retrieve encryption keys from its QKD device. Therefore, Quantum networks securely communicate encryption keys always.

Figure 15: Point-to -Point MACsec Link Encryption using SKIP



Prerequisites

- Configure MACsec Pre-Sared Key (PSK). For more information, see MACsec PSK, on page 201.
- Configure MACsec in the PPK mode.
- An external QKD devices network.
- To establish secured connection (https) from the router to the QKD server, you must import the same root CA which signed the QKD server also. For this, you must:
  - Add the QKD server CA to the trustpoint in the router. For more information, see Configure Trustpoint.
  - Import the QKD server root CA certificate in the router. For more information, see Configure Certificate Enrollment Using Cut-and-Paste.

Configuration

The following example details how to establish Point to Point MACsec Link Encryption using SKIP:

Router 1:

1. Configure the QKD profile.

```
Router# config
Router(config)# sks profile ProfileR1toR2 type remote
Router(config-sks-profile)# kme server ipv4 192.0.2.34 port 10001
Router(config-sks-profile)# commit
```

2. Map the QKD profile to the MACsec policy.

```
Router# config
Router(config)# macsec-policy R1toR2
Router(config-macsec-policy)# ppk sks-profile ProfileR1toR2
Router(config-macsec-policy)# commit
```

3. Apply MACsec policy to the interfaces.

```
Router# config
Router(config)#interface hundredGigE 0/1/0/10
Router(config-if)#ipv4 address 192.0.2.1 255.255.255.0
Router(config-if)#macsec psk-keychain mac_chain policy R1toR2
Router(config)#commit
Router(config)#interface hundredGigE 0/1/0/11
Router(config-if)#ipv4 address 192.0.3.1 255.255.255.0
Router(config-if)#macsec psk-keychain mac_chain policy R1toR2
Router(config)#commit
Router(config)#interface hundredGigE 0/1/0/12
Router(config-if)#ipv4 address 192.0.4.1 255.255.255.0
Router(config-if)#macsec psk-keychain mac_chain policy R1toR2
Router(config)#commit
Router(config)#interface hundredGigE 0/1/0/9
Router(config-if)#ipv4 address 192.0.5.1 255.255.255.0
Router(config-if)#macsec psk-keychain mac_chain policy R1toR2
Router(config)#commit
```

Router 2:

1. Configure the QKD profile.

```
Router#config
Router(config)#sks profile ProfileR2toR1 type remote
Router(config-sks-profile)#kme server ipv4 192.0.2.35 port 10001
Router(config-sks-profile)#commit
```

2. Map the QKD profile to the MACsec policy.

```
Router#config
Router(config)#macsec-policy R2toR1
Router(config-macsec-policy)#ppk sks-profile ProfileR2toR1
Router(config-macsec-policy)#commit
```

3. Apply MACsec policy to the interfaces.

```
Router#config
Router(config)#interface hundredGigE 0/1/0/10
Router(config-if)#ipv4 address 192.0.2.2 255.255.255.0
Router(config-if)#macsec psk-keychain mac_chain policy R2toR1
Router(config-if)#commit
Router(config)#interface hundredGigE 0/1/0/11
Router(config-if)#ipv4 address 192.0.3.2 255.255.255.0
Router(config-if)#macsec psk-keychain mac_chain policy R2toR1
Router(config-if)#commit
Router(config)#interface hundredGigE 0/1/0/12
Router(config-if)#ipv4 address 192.0.4.2 255.255.255.0
Router(config-if)#macsec psk-keychain mac_chain policy R2toR1
Router(config-if)#commit
Router(config)#interface hundredGigE 0/1/0/9
Router(config-if)#ipv4 address 192.0.5.2 255.255.255.0
Router(config-if)#macsec psk-keychain mac_chain policy R2toR1
Router(config-if)#commit
```

Running Configuration

Router 1:

```
sks profile ProfileR1toR2 type remote
 kme server ipv4 192.0.2.34 port 10001
!
macsec-policy R1toR2
 ppk
  sks-profile ProfileR1toR2
 !
!
interface hundredGigE 0/1/0/10
 ipv4 address 192.0.2.1 255.255.255.0
 macsec psk-keychain mac_chain policy R1toR2
!
interface hundredGigE 0/1/0/11
 ipv4 address 192.0.3.1 255.255.255.0
 macsec psk-keychain mac_chain policy R1toR2
!
interface hundredGigE 0/1/0/12
 ipv4 address 192.0.4.1 255.255.255.0
 macsec psk-keychain mac_chain policy R1toR2
!
interface hundredGigE 0/1/0/9
 ipv4 address 192.0.5.1 255.255.255.0
 macsec psk-keychain mac_chain policy R1toR2
!
```

Router 2:

```
sks profile ProfileR2toR1 type remote
 kme server ipv4 192.0.2.35 port 10001
!
macsec-policy R2toR1
 ppk
  sks-profile ProfileR2toR1
 !
!
interface hundredGigE 0/1/0/10
 ipv4 address 192.0.2.2 255.255.255.0
 macsec psk-keychain mac_chain policy R2toR1
!t
interface hundredGigE 0/1/0/11
 ipv4 address 192.0.3.2 255.255.255.0
 macsec psk-keychain mac_chain policy R2toR1
!
interface hundredGigE 0/1/0/12
 ipv4 address 192.0.4.2 255.255.255.0
 macsec psk-keychain mac_chain policy R2toR1
!
interface hundredGigE 0/1/0/9
 ipv4 address 192.0.5.2 255.255.255.0
 macsec psk-keychain mac_chain policy R2toR1
!
```

Verification

```
Router(ios)# show crypto sks profile all
Profile Name         :ProfileR1toR2
Myidentifier        :Router1
Type                :Remote
Reg Client Count     :1
```

```
Server
IP                  :192.0.2.34
Port                :10001
Vrf                 :Notconfigured
Source Interface    :Notconfigured
Status              :Connected
Entropy             :true
Key                 :true
Algorithm           :QKD
Local identifier    :Alice
Remote identifier   :Alice, Bob

Peerlist
QKD ID               :Alice
State                :Connected

QKD ID               :Bob
State                :Connected


Router(ios)# show crypto sks profile all stats
Profile Name            : ProfileR1toR2
My identifier           : Router1
Server
 IP                     : 192.0.2.34
 Port                   : 10001
 Status                 : connected
Counters
 Capability request         : 1
 Key request                : 3
 Key-id request             : 0
 Entropy request            : 0
 Capability response        : 1
 Key response               : 3
 Key-id response            : 0
 Entropy response           : 0
 Total request              : 4
 Request failed             : 0
 Request success            : 4
 Total response             : 4
 Response failed            : 0
 Response success           : 4
 Retry count                : 0
 Response Ignored           : 0
 Cancelled count            : 0
Response time
 Max Time                   : 100 ms
 Avg Time                   : 10  ms
 Min Time                   : 50  ms
Last transaction
 Transaction Id             : 9
 Transaction type           : Get key
 Transaction status         : Response data received, successfully
 Http code                  : 200 OK (200)
```

Global MACsec Shutdown

The MACsec shutdown feature allows administrator to disable MACsec and re-enable it without modifying the existing MACsec configuration.

Enabling the **macsec shutdown** command, brings down all MACsec sessions on the MACsec-enabled interfaces and resets ports to non-macsec mode. The already existing MACsec configurations remain unaffected by enabling this feature.

Disabling the **macsec shutdown** command, brings up macsec sessions for the configured interfaces and enforces MACsec policy on the port. This feature is disabled by default.

Configure MACsec Shutdown

The following configuration enables the MACsec shutdown on a chassis:

```
RP/0/RP0/CPU0:router# configure terminal
RP/0/RP0/CPU0:router(config)# macsec shutdown
```

Warning Configuring **macsec shutdown** command disables MACsec on all data ports, system wide. Execute **clear** command to erase cached configuration or **commit** command to continue.

Verify MACsec Shutdown

The **show macsec mka session** command displays a shutdown banner indicating that the MACsec shutdown is enabled.

```
RP/0/RP0/CPU0:router# show macsec mka session
Fri Apr 13 11:56:57.409 IST

********** MACsec shutdown enabled **********
```

The **show macsec mka interface detail** command displays a shutdown banner and the interface-related information.

```
RP/0/RP0/CPU0:fretta-2#show macsec mka interface detail
Fri Apr 13 11:57:02.685 IST
********** MACsec shutdown enabled **********

Number of interfaces on node node0_3_CPU0 : 1
--------------------------------------------------

    Interface Name           : HundredGigE0/3/0/8
    Interface Namestring     : HundredGigE0/3/0/8
    Interface short name     : Hu0/3/0/8
    Interface handle         : 0x1800170
    Interface number         : 0x1800170
    Interface MAC            : 008a.9622.a9d0
    Ethertype                : 888E
    MACsec Shutdown          : TRUE
    Config Received          : TRUE
    IM notify Complete       : TRUE
    Interface CAPS Add       : FALSE
    RxSA CAPS Add            : FALSE
    TxSA CAPS Add            : FALSE
    MKA PSK Info
      Key Chain Name         : kc1
      MKA Cipher Suite       : AES-256-CMAC
      CKN                    : 12 34 56
    MKA fallback_PSK Info
```

```
    fallback keychain Name : fb1
    MKA Cipher Suite       : AES-256-CMAC
    CKN                    : ff ff ff
  Policy                   : *DEFAULT POLICY*
```

Syslog Messages for MACsec Shutdown

The following syslog messages are generated when MACsec shutdown is enabled.

```
%L2-MKA-5-MACSEC_SHUTDOWN_ENABLED : Shutdown ON, disable MACsec on all MACsec enabled ports
%L2-MKA-5-SESSION_STOP            : (Hu0/3/0/8) MKA session stopped,
CKN                               : 123456
%L2-MKA-4-SESSION_UNSECURED       : (Hu0/3/0/8) MKA Session was stopped and is not secured,

CKN                               :123456
%L2-MKA-5-MACSEC_DISABLED         : (Hu0/3/0/8), MACsec disabled (shutdown ON)
```

The following syslog messages are generated when MACsec shutdown is disabled.

```
%L2-MKA-5-MACSEC_SHUTDOWN_DISABLED : Shutdown OFF, resume MACsec on all MACsec enabled ports
%L2-MKA-5-MACSEC_ENABLED                    : (Hu0/3/0/8), MACsec enabled with MUST_SECURE
%L2-MKA-5-SESSION_START                     : (Hu0/3/0/8) MKA session started
CKN                                         :  123456
%L2-MKA-6-MKPDU_ICV_SUCCESS                 : (Hu0/3/0/8), ICV verification success for
RxSCI(008a.9600.60b0/0001), CKN(123456)
%L2-MKA-6-FALLBACK_PSK_MKPDU_ICV_SUCCESS    : (Hu0/3/0/8), ICV verification success for
RxSCI(008a.9600.60b0/0001), CKN(FFFFFF)
%L2-MKA-5-SESSION_SECURED                   : (Hu0/3/0/8) MKA session secured
 CKN                                        :  123456
```

MACsec ISSU

The Cisco IOS XR Software supports in-service software upgrade (ISSU) for Media Access Control Security (MACsec) on the 64-bit IOS XR operating system. This feature allows you to upgrade the network systems without interrupting the secure data connectivity provided by the MACsec session. Such upgrades are feasible if the system and each of its peers support in-service software upgrade.

Commands introduced are:

- suspendFor
- suspendOnRequest

The MACsec ISSU feature is implemented as per the IEEE compliance standard, IEEE Std 802.1Xbx™-2014. It works by suspending the MACsec Key Agreement (MKA) protocol operation temporarily during the ISSU. Once the control plane operation is suspended, the data plane continues to do the encryption with the MACsec hardware keys that are already programmed.

Supported Hardware for MACsec ISSU

The MACsec ISSU feature is supported on Cisco ASR 9000 High Density 100GE Ethernet line cards. The supported hardware variants are:

- A9K-4X100GE-SE
- A9K-8X100GE-SE

- A9K-MPA-1X100GE
- A9K-MPA-2X100GE
- A9K-MPA-20X10GE
- A9K-400G-DWDM-TR

Restrictions for MACsec ISSU

These restrictions apply to MACsec ISSU feature:

- Supported only on 64-bit IOS XR operating system, and on specific hardware (listed in previous section)
- Supported only on pre-shared keys (PSK) based MACsec; not on Extensible Authentication Protocol (EAP) based MACsec. The system terminates the ISSU process if any of the interfaces has EAP MACSec configuration.
- The MACsec ISSU is not supported from release version lower than Cisco IOS XR Software Release 7.1.1 to versions higher or equal toRelease 7.1.1.



Note Disable the MACSec on interfaces or configure **macsec shutdown** command at global configuration mode (if applicable) to run a successful ISSU on the software with release versions lower than Release 7.1.1.

- ISSU is supported only for MACSec sessions running on extended packet numbering (xpn) cipher suites (GCM-AES-XPN-128 and GCM-AES-XPN-256). The system terminates ISSU if there are sessions with non-xpn cipher suites (GCM-AES-128 or GCM-AES-256). The key server selects the cipher suite; the configuration of non-key server cipher suite is insignificant.
- The system terminates MACsec ISSU if there are sessions which are not yet in **suspended** state (use the **show macsec mka session** command to view the session state) after 30 seconds of the load execution phase of ISSU.

Options to Control MKA Protocol Suspension Initiation for ISSU

You can use these two commands under the macsec policy configuration mode to control MKA protocol suspension initiation:

- **suspendFor**: Initiates suspension if it is the key server or requests suspension if it is the non-key server. This option helps admins to control the network by preventing software upgrades that the system triggers without the permission of the key server.
- **suspendOnRequest**: Initiates suspension if it is the key server and when another participant has requested for suspension.

By default, the system enables both options.

Command Usage	Action on the Key Server	Action on the Non-Key Server
suspendFor disable	Disables MKA suspension initiation	Disables the request for MKA suspension
suspendOnRequest disable	Rejects the MKA suspension request from the non-key server	Not applicable

Configuration Example

```
Router#configure
Router(config)#macsec-policy test-policy-mp
/* Disables MKA suspension initiation (if it is the key server) or
disables the request for MKA suspension (if it is the non-key server) */
Router(config-macsec-policy)#suspendFor disable

/* Disables any MKA suspension request from the non-key server */
Router(config-macsec-policy)#suspendOnRequest disable
```

Running Configuration

```
!
macsec-policy test-policy-mp
 suspendFor disable
 suspendOnRequest disable
!
end
```

Verification

A new session state, **SUSPENDED**, is introduced to display the status of MKA suspension operation during ISSU.

```
Router#show macsec mka session
Mon Apr  1 13:13:43.334 IST

NODE: node0_1_CPU0
==========================================================================================

   Interface-Name        Local-TxSCI       #Peers   Status   Key-Server   PSK/EAP      CKN

==========================================================================================

     Hu0/1/0/0       0201.9ab0.85af/0001     1     Suspended     YES      PRIMARY     1234
```

You can use this command to see the details of the MACsec policy:

```
Router#show macsec policy detail
Tue May 21 14:19:31.101 IST
Total Number of Policies = 2
-------------------------------------------------------
Policy Name               : *DEFAULT POLICY*
      Cipher Suite            : GCM-AES-XPN-256
```

```
      Key-Server Priority   : 16
      Window Size           : 64
      Conf Offset           : 0
      Replay Protection     : TRUE
      Delay Protection      : FALSE
      Security Policy       : Must Secure
      Vlan Tags In Clear    : 1
      LACP In Clear         : FALSE
      Sak Rekey Interval    : OFF
      Include ICV Indicator :  FALSE
      Use Eapol PAE in ICV  : FALSE
      Suspend On Request    : Enabled
      Suspend For           : Enabled

Policy Name                 : test-policy-mp
      Cipher Suite          : GCM-AES-XPN-256
      Key-Server Priority   : 16
      Window Size           : 64
      Conf Offset           : 0
      Replay Protection     : TRUE
      Delay Protection      : FALSE
      Security Policy       : Must Secure
      Vlan Tags In Clear    : 1
      LACP In Clear         : FALSE
      Sak Rekey Interval    : OFF
      Include ICV Indicator : FALSE
      Use Eapol PAE in ICV  : FALSE
     Suspend On Request     : Disabled
     Suspend For            : Disabled
```

You can use the **Suspended Peer List** field in the **show macsec mka session detail** command to view the list of peers of the key server that had requested for suspension.

```
Router#show macsec mka session detail
Mon Apr  1 13:13:45.893 IST
NODE: node0_1_CPU0
 MKA Detailed Status for MKA Session
==================================
Status: SUSPENDED - Secured MACsec with suspended MKA operations

Local Tx-SCI                    : 0201.9ab0.85af/0001
Local Tx-SSCI                   : 2
Interface MAC Address           : 0201.9ab0.85af
MKA Port Identifier             : 1
Interface Name                  : Hu0/1/0/0
CAK Name (CKN)                  : 1234
CA Authentication Mode          : PRIMARY-PSK
Keychain                        : kc1
Member Identifier (MI)          : 89E20E40ACED97317596CCC0
Message Number (MN)             : 156
Authenticator                   : NO
Key Server                      : YES
MKA Cipher Suite                : AES-256-CMAC
Configured MACSec Cipher Suite  : GCM-AES-XPN-256

Latest SAK Status               : Rx & Tx
Latest SAK AN                   : 2
Latest SAK KI (KN)              : 89E20E40ACED97317596CCC000000001 (1)
Old SAK Status                  : No Rx, No Tx
Old SAK AN                      : 1
Old SAK KI (KN)                 : RETIRED (0)
```

```
SAK Transmit Wait Time          : 0s (Not waiting for any peers to respond)
SAK Retire Time                 : 0s (No Old SAK to retire)
Time to SAK Rekey               : NA
Time to exit suspension         : 120s

MKA Policy Name                 : *DEFAULT POLICY*
Key Server Priority             : 16
Delay Protection                : FALSE
Replay Window Size              : 64
Include ICV Indicator           : FALSE
Confidentiality Offset          : 0
Algorithm Agility               : 80C201
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 Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------
DA057FA6983845205FD0EB28      162       0257.3fae.5cda/0001    1        16

Potential Peer List:
--------------------------------------------------------------------------------
           MI                  MN             Rx-SCI          SSCI  KS-Priority
--------------------------------------------------------------------------------

Suspended Peer List:
--------------------------------------------------------------------------------
       Rx-SCI          SSCI
--------------------------------------------------------------------------------
  02573fae5cda0001      1
Peers Status:
Last Tx MKPDU           : 2019 Apr 01 13:13:45.350
Peer Count              : 1

RxSCI                   : 02573FAE5CDA0001
  MI                    : DA057FA6983845205FD0EB28
  Peer CAK              : Match
  Latest Rx MKPDU       : 2019 Apr 01 13:13:44.238
```

Also, these SYSLOGS indicate various stages of the ISSU process on the key server and the non-key server:

- **L2-MKA-5-SUSPENSION-REQUESTED**
  - On the non-key server—when it requests for suspension. (ISSU)

```
(Hu0/1/0/0), Requesting suspension of MACsec control plane operation
```

- **L2-MKA-5-SUSPENSION-START-REQUEST_RECEIVED**
  - On the key server—when it receives non-zero value for the suspendFor parameter from the non-key server. The key server accepts or rejects the suspension request based on the value configured for the **suspendOnRequest** command.

```
(Hu0/1/0/0), MACsec control plane operation suspension start request from
```

```
Peer(02573fae5cda0001) accepted.
```

or

```
(Hu0/1/0/0), MACsec control plane operation suspension start request from
Peer(02573fae5cda0001) rejected (policy conflict).
```

- **L2-MKA-5-SUSPENSION-START**
  - On the key server—when it initiates suspension.
  - On the non-key server—when it receives non-zero value for the **suspendFor** parameter from the key server.

```
(Hu0/1/0/0), MACsec control plane operation suspended.
```

- **L2-MKA-5-SUSPENSION-STOP-REQUEST_RECEIVED**
  - On the key server—when it receives a zero value for the**suspendFor** parameter from the peer which had previously requested for suspension.

```
(Hu0/1/0/0), MACsec control plane operation suspension stop received from
Peer(02573fae5cda0001)
```

- **L2-MKA-5-SUSPENSION-STOP**
  - On the key server—when it terminates the suspension.
  - On the non-key server—when it receives a zero value for the **suspendFor** parameter from the key server.

```
(Hu0/1/0/0), MACsec control plane operation resumed
```

Related Topics

MACsec ISSU, on page 286

Associated Commands

- suspendFor
- suspendOnRequest

MACsec SNMP MIB (IEEE8021-SECY-MIB)

Table 37: Feature History Table

Feature Name	Release Information	Description
MACsec SNMP MIB (IEEE8021-SECY-MIB) support on A99-10X400GE-X-SE	Release 7.5.3	With this feature, we have enabled IEEE8021-SECY-MIB capabilities on the A99-10X400GE-X-SE line card. The IEEE8021-SECY-MIB enables the user to query on the SecY (Security Entity) data, encryption, decryption, and hardware statistics.

The IEEE8021-SECY-MIB provides Simple Network Management Protocol (SNMP) access to the MAC security entity (SecY) MIB running with IOS XR MACsec-enabled line cards. The IEEE8021-SECY-MIB is used to query on the SecY data, encryption, decryption, and the hardware statistics. The SecY is an entity that operates the MAC Security protocol in the router. The SecY MIB data is queried only on the Controlled Port.

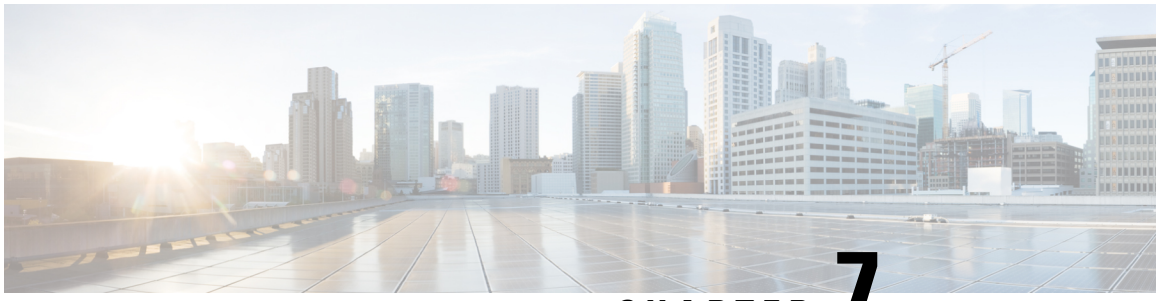
The object ID of the IEEE8021-SECY-MIB is 1.0.8802.1.1.3. The IEEE8021-SECY-MIB contains the following tables that specifies the detailed attributes of the MACsec Controlled Port interface index.

Table 38: IEEE8021-SECY-MIB Table

Tables	OID
secyIfTable	1.0.8802.1.1.3.1.1.1
secyTxSCTable	1.0.8802.1.1.3.1.1.2
secyTxSATable	1.0.8802.1.1.3.1.1.3
secyRxSCTable	1.0.8802.1.1.3.1.1.4
secyRxSATable	1.0.8802.1.1.3.1.1.5
secyCipherSuiteTable	1.0.8802.1.1.3.1.1.6
secyTxSAStatsTable	1.0.8802.1.1.3.1.2.1
secyTxSCStatsTable	1.0.8802.1.1.3.1.2.2
secyRxSAStatsTable	1.0.8802.1.1.3.1.2.3
secyRxSCStatsTable	1.0.8802.1.1.3.1.2.4
secyStatsTable	1.0.8802.1.1.3.1.2.5

References:

- For more technical details on MACsec SNMP MIB (IEEE8021-SECY-MIB) support in ASR 9000 Series Routers, download the *IEEE8021-SECY-MIB* from MIB Locator in Cisco Feature Navigator.
- For more information on the Simple Network Management (SNMP) Protocol, see chapter *Configuring Simple Network Management Protocol* in *System Management Configuration Guide for Cisco 8000 Series Routers*.
- For more information on the IEEE8021-SecY-MIB, see the following URL:http://www.ieee802.org/1/files/public/MIBs/IEEE8021-SECY-MIB-200601100000Z.mib



CHAPTER 7

Implementing Type 6 Password Encryption

Type 6 password encryption uses a reversible 128-bit AES encryption algorithm for storing passwords. Type 6 password encryption allows secure, and encrypted storage of plain-text passwords on the device. The device can decrypt the encrypted passwords into their original plain-text format.

You can use Type 6 password encryption to securely store plain text key strings for authenticating BGP, IP SLA, IS-IS, MACsec, OSPF, and RSVP sessions.

Feature History for Implementing Type 6 Password Encryption

Release	Modification
Release 7.0.1	This feature was introduced.

- Restrictions for Type 6 password encryption , on page 295
- How to Implement Type 6 Password Encryption , on page 295

Restrictions for Type 6 password encryption

Type 6 encryption is not supported for SNMPv3, NTP, and LDP. If Type 6 encryption is enabled globally, these protocols continue to use Type 7 encryption even when Type 6 is configured.

How to Implement Type 6 Password Encryption

Scenario - The following 3-step process explains the Type 6 password encryption process for authenticating BGP sessions between two routers, R1 and R2.

Follow the first two steps for all Type 6 password encryption scenarios. The third step, *Creating BGP Sessions*, is specific to BGP. Similarly, you can enable Type 6 password encryption for OSPF, IS-IS, or other protocol sessions. For details on creating these protocol sessions, see the content in *Configure>Routing* listed here. For MACsec authentication, refer the *Configure MACsec* chapter.

Enabling Type6 Feature and Creating a Primary Key (Type 6 Server)

The Type6 encryption key, hereafter referred to as primary key in this chapter, is the password or key that encrypts all plain text key strings in the router configuration. An Advance Encryption Standard (AES) symmetric

cipher does the encryption. The router configuration does not store the primary key. You cannot see or access the primary key when you connect to the router.

Creating the Primary Key

Use the **key config-key password-encryption** command to create the primary key.

Configuration Example

```
R1 & R2 # key config-key password-encryption

Fri Jul 19 12:22:45.519 UTC
New password Requirements: Min-length 6, Max-length 64
Characters restricted to [A-Z][a-z][0-9]
Enter new key :
Enter confirm key :
Master key operation is started in background
```

Once the command is executed, the **Master key operation**—creating, updating, or deleting the primary key—happens in the background. You can use the **show type6 server** command to view the status of the primary key operation.

When the key is created, it is stored internally; not as part of the router configuration. The router does not display the primary key as part of the running configuration. So, you cannot see or access the primary key when you connect to the router.

Enabling Type 6 Password Encryption

```
/* Enable Type 6 password encryption */
R1 & R2 (config)# password6 encryption aes
R1 & R2 (config)# commit
Fri Jul 19 12:22:45.519 UTC
```

Modifying the Primary Key

Note The Type 6 primary key update results in configuration change of the key chain and the other clients using Type 6. As the failure of router being configured can disrupt the product network, it is recommended to perform the primary key update operation during a maintenance window. Else, routing protocol sessions might fail.

The primary key is not saved to the running configuration, but the changes are persistent across reloads. The primary key update cannot be rolled back. That is, once the primary key is modified, you cannot revert to the older key using the **rollback configuration** command.

Enter the **key config-key password-encryption** command, and the old key and new key information.

```
R1 & R2# key config-key password-encryption

New password Requirements: Min-length 6, Max-length 64
Characters restricted to [A-Z][a-z][0-9]
Enter old key :
Enter new key :
Enter confirm key :
Master key operation is started in background
```

Deleting the Primary Key

```
R1 & R2# configure
R1 & R2 (config)# no password6 encryption aes
R1 & R2 (config)# commit
R1 & R2 (config)# exit
R1 & R2# key config-key password-encryption delete

WARNING: All type 6 encrypted keys will become unusable
Continue with master key deletion ? [yes/no]:yes
Master key operation is started in background
```

Verification

Verify that the primary key configuration and Type 6 feature configuration state are in the *Enabled* state. The **Master key Inprogress** field displays **No** to indicate that the primary key activity is complete (created, modified, or deleted). When you disable a primary key, **Disabled** is displayed for all the three states.

```
R1 & R2#show type6 server

Fri Jul 19 12:23:49.154 UTC
Server detail information:
==========================================
AES config State       :       Enabled
Masterkey config State :       Enabled
Type6 feature State    :       Enabled
Master key Inprogress  :       No
Masterkey Last updated/deleted : Mon Nov 4 15:55:11 2024
Masterkey Size (bits) : 48
Masterkey Hash (Algo:sha512,Format:base64) :
ujJTh2rta8ItSm/1PYQGxq2GQZXtFEq1yHYhtsIztUi66uaVbfNG7IwX9eoQ817jy8UUeX7X3dMUVGTioLq0Ew==
```

Verify Type 6 trace server details.

```
R1 & R2#show type6 trace server all

Fri Jul 19 12:26:05.111 UTC
Client file lib/type6/type6_server_wr
25 wrapping entries (18496 possible, 64 allocated, 0 filtered, 25 total)
Jul 19 09:59:27.168 lib/type6/type6_server_wr 0/RP0/CPU0 t7145 ***** Type6 server process
started Respawn count (1) ****
…
…
Jul 19 12:22:59.908 lib/type6/type6_server_wr 0/RP0/CPU0 t7145 User has started Master key
 operation (CREATE)
Jul 19 12:22:59.908 lib/type6/type6_server_wr 0/RP0/CPU0 t7145 Created Master key in TAM
successfully
Jul 19 12:23:00.265 lib/type6/type6_server_wr 0/RP0/CPU0 t7145 Master key Available set to
 (AVAILABLE)
Jul 19 12:23:00.272 lib/type6/type6_server_wr 0/RP0/CPU0 t7145 Master key inprogress set
to (NOT INPROGRESS)
```

From Cisco IOS XR Software Release 7.0.2 and later, you can use the **show type6 masterkey update status** command to display the update status of the primary key. Prior to this release, you could use the **show type6 clients** command for the same purpose.

```
Router#show type6 masterkey update status
Thu Sep 17 06:48:56.595 UTC
Type6 masterkey operation is NOT inprogress


Router#show type6 masterkey update status
```

```
Thu Sep 17 06:50:07.980 UTC
Type6 masterkey operation is inprogress

Masterkey upate status information:
Client Name                Status
===================================
keychain                   INPROGRESS
```

Clear Type 6 Client State

You can use the **clear type6 client** command in EXEC mode to clear the Type 6 client state.

If the primary key update operation is stuck at any stage, then you can use this **clear** command to clear that state. You can track the primary key update operation using the **show type6 server** command output. If the *Master key Inprogress* field in that output displays as *YES*, then you can use **show type6 masterkey update status** command (or, **show type6 clients** command, prior to Release 7.0.2) to check which client has not completed the operation. Accordingly, you can clear that particular client using the **clear** command.

Associated Commands

- **clear type6 client**
- **key password-encryption**
- **password6 encryption aes**
- **show type6**

Implementing Key Chain for BGP Sessions (Type 6 Client)

For detailed information on key chains, refer the *Implementing Keychain Management* chapter.

If you enable Type 6 password encryption, plain-text keys are encrypted using Type 6 encryption. Enter plain-text key-string input in alphanumeric form. If you enable MACsec with Type 6 password encryption, the key-string input is in hexadecimal format.

Configuration

```
/* Enter the key chain details */
R1 & R2# configure
R1 & R2(config)# key chain my-test-keychain
R1 & R2(config-type6_password)# key 1
```

Enter the Type 6 encrypted format using the **key-string password6** command.

Note Using the **key-string** command, you can enter the password in clear text format or Type 6 encrypted (already encrypted password) format, as used in this scenario.

Note Enable the same key string for all the routers.

```
R1 & R2 (config-type6_password-1)# key-string password6
666449644369554484a4448674b695e685d56565d676364554b64555f4c5c645b
R1 & R2 (config-type6_password-1)# cryptographic-algorithm HMAC-MD5
```

```
R1 & R2 (config-type6_password-1)# accept-lifetime 1:00:00 october 24 2005 infinite
R1 & R2 (config-type6_password-1)# send-lifetime 1:00:00 october 24 2005 infinite
R1 & R2 (config-type6_password-1)# commit
```

Note Border Gateway Protocol (BGP) supports only HMAC-MD5 and HMAC-SHA1-12.

Verification

Verify key chain trace server information.

```
R1 & R2# show key chain trace server both

Sat Jul 20 16:44:08.768 UTC
Client file lib/kc/kc_srvr_wr
4 wrapping entries (18496 possible, 64 allocated, 0 filtered, 4 total)
Jul 20 16:43:26.342 lib/kc/kc_srvr_wr 0/RP0/CPU0 t312 *********kc_srvr process
started*********
Jul 20 16:43:26.342 lib/kc/kc_srvr_wr 0/RP0/CPU0 t312 (kc_srvr) Cerrno DLL registration
successfull
Jul 20 16:43:26.349 lib/kc/kc_srvr_wr 0/RP0/CPU0 t312 (kc_srvr) Initialised sysdb connection
Jul 20 16:43:26.612 lib/kc/kc_srvr_wr 0/RP0/CPU0 t317 (kc_srvr_type6_thread) Succesfully
registered as a type6 client
```

Verify configuration details for the key chain.

```
R1 & R2# show key chain type6_password
Sat Jul 20 17:05:12.803 UTC

Key-chain: my-test-keychain -
  Key 1 -- text "6664496443695544484a4448674b695e685d56565d676364554b64555f4c5c645b"
    Cryptographic-Algorithm -- HMAC_MD5
    Send lifetime --  01:00:00, 24 Oct 2005 - Always valid  [Valid now]
    Accept lifetime -- 01:00:00, 24 Oct 2005 - Always valid [Valid now]
```

Associated Commands

- **key chain**
- **key-string password6**
- **show key chain trace server both**

Creating a BGP Session (Type 6 Password Encryption Use Case)

This example provides iBGP session creation configuration. To know how to configure the complete iBGP network, refer the *BGP Configuration Guide* for the platform.

Configuration Example

```
/* Create BGP session on Router1 */
R1# configure
R1(config)# router bgp 65537
```

Ensure that you use the same key chain name for the BGP session and the Type 6 encryption (for example, *my-test-keychain* in this scenario).

```
R1 (config-bgp)# neighbor 10.1.1.11 remote-as 65537
R1 (config-bgp)# keychain my-test-keychain
R1 (config-bgp)# address-family ipv4 unicast
R1 (config-bgp)# commit
```

Repeat the above steps on Router 2 as well.

Ensure that you use the same session and keychain for all the routers (R1 and R2 in this case).

```
/* Create BGP session on Router2 */
R2 (config)# router bgp 65537
R2 (config-bgp)# neighbor 10.1.1.1 remote-as 65537
R2 (config-bgp)# keychain my-test-keychain
R2 (config-bgp)# address-family ipv4 unicast
R2 (config-bgp)# commit
```

Verification

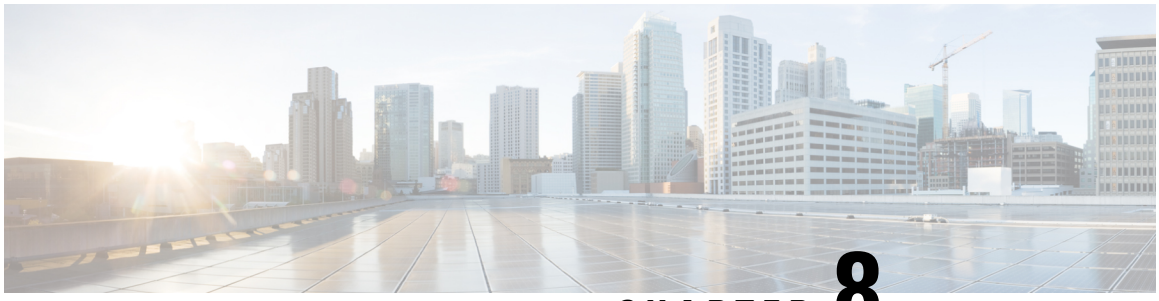
On the routers R1 and R2, verify that the BGP NBR state is in the *Established* state.

```
R1# show bgp sessions
Neighbor      VRF      Spk    AS      InQ  OutQ  NBRState     NSRState
10.1.1.11     default  0      65537   0    0     Established  None
```

```
R2# show bgp sessions
Neighbor      VRF      Spk    AS      InQ  OutQ  NBRState     NSRState
10.1.1.1     default  0      65537   0    0     Established  None
```

Associated Commands

- **Session-group**
- **show BGP sessions**



CHAPTER 8

Implementing Lawful Intercept

Lawful intercept is the process by which law enforcement agencies conduct electronic surveillance of circuit and packet-mode communications, authorized by judicial or administrative order. Service providers worldwide are legally required to assist law enforcement agencies in conducting electronic surveillance in both circuit-switched and packet-mode networks.

Only authorized service provider personnel are permitted to process and configure lawfully authorized intercept orders. Network administrators and technicians are prohibited from obtaining knowledge of lawfully authorized intercept orders, or intercepts in progress. Error messages or program messages for intercepts installed in the router are not displayed on the console.

Lawful Intercept is not a part of the Cisco IOS XR software by default. You have to install it separately by installing and activating **asr9k-li-px.pie** .

Feature History for Implementing Lawful Intercept

Release	Modification
Release 4.1.0	This feature was introduced.
Release 4.2.0	High Availability support for Lawful Intercept was added. Support for IPv6 Lawful Intercept was added.
Release 4.3.2	Lawful Intercept is available as a separate package. It is no longer a part of the Cisco IOS XR software.
Release 7.0.1	Overlapping tap functionality is made available on Cisco ASR 9000 4th Generation QSFP28 based dense 100GE line cards (A9K-8X100GE-X-TR, A9K-16X100GE-TR, A9K-32X100GE-TR and A99-16x100-X-SE) as well.

- Prerequisites for Implementing Lawful Intercept, on page 302
- Restrictions for Implementing Lawful Intercept, on page 303
- Information About Lawful Intercept Implementation, on page 304
- Intercepting IPv4 and IPv6 Packets, on page 308
- High Availability for Lawful Intercept, on page 311
- Installing Lawful Intercept (LI) Package, on page 312
- How to Configure SNMPv3 Access for Lawful Intercept, on page 317
- Configuration Example for Inband Management Plane Feature Enablement, on page 322
- Increased Lawful Interception rates, on page 322

• Additional References, on page 324

Prerequisites for Implementing Lawful Intercept

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Lawful intercept implementation also requires that these prerequisites are met:

- Cisco ASR 9000 Series Aggregation Services Router will be used as content Intercept Access Point (IAP) router in lawful interception operation.
- **Provisioned router—**The router must be already provisioned. For more information, see *Cisco ASR 9000 Series Aggregation Services Router Getting Started Guide*.

Tip For the purpose of lawful intercept taps, provisioning a loopback interface has advantages over other interface types.

- **Understanding of SNMP Server commands in Cisco IOS XR software—**Simple Network Management Protocol, version 3 (SNMP v3), which is the basis for lawful intercept enablement, is configured using commands described in the module *SNMP Server Commands* in *System Management Command Reference for Cisco ASR 9000 Series Routers*. To implement lawful intercept, you must understand how the SNMP server functions. For this reason, carefully review the information described in the module *Implementing SNMP* in *System Management Configuration Guide for Cisco ASR 9000 Series Routers*.
- **Lawful intercept must be explicitly disabled—**It is automatically enabled on a provisioned router. However, you should not disable LI if there is an active tap in progress, because this deletes the tap.
- **Management plane configured to enable SNMPv3—**Allows the management plane to accept SNMP commands, so that the commands go to the interface (preferably, a loopback) on the router. This allows the mediation device (MD) to communicate with a physical interface.
- **VACM views enabled for SNMP server—**View-based access control model (VACM) views must be enabled on the router.
- **Provisioned MD—**For detailed information, see the vendor documentation associated with your MD. For a list of MD equipment suppliers preferred by Cisco, see http://www.cisco.com/en/US/tech/tk583/tk799/tsd_technology_support_protocol_home.html.
- **VoIP surveillance-specific requirements**
  - **Lawful-intercept-enabled call agent—**A lawful-intercept-enabled call agent must support interfaces for communications with the MD, for the target of interest to provide signaling information to the MD. The MD extracts source and destination IP addresses and Real-Time Protocol (RTP) port numbers from the Session Description Protocol (SDP) signaling information for the target of interest. It uses these to form an SNMPv3 SET, which is sent to the router acting as the content IAP to provision the intercept for the target of interest.

    The MD uses the CISCO-TAP2-MIB to set up communications between the router acting as the content IAP, and the MD.

The MD uses the CISCO-IP-TAP-MIB to set up the filter for the IP addresses and port numbers to be intercepted and derived from the SDP.

- Routers to be used for calls by the target number must be provisioned for this purpose through the MD.
- The MD that has been provisioned with the target number to be intercepted.

- **Data session surveillance-specific requirements**
  - Routers to be used by the data target that have been provisioned for this purpose through the MD.
  - **The MD that has been provisioned with the user login ID, mac address of the user CPE device, or the DSLAM physical location ID—**The IP address is the binding that is most frequently used to identify the target in the network. However, alternative forms of information that uniquely identify the target in the network might be used in some network architectures. Such alternatives include the MAC address and the acct-session-id.

- The MD can be located anywhere in the network but must be reachable from the content IAP router, which is being used to intercept the target. MD should be reachable ONLY from global routing table and NOT from VRF routing table.

Restrictions for Implementing Lawful Intercept

The following restrictions are applicable for Lawful Intercept:

- If lawful intercept is set up separately for two inter-communicating hosts with two different mediation devices, then by default, only the ingress traffic on the ASR 9000 router from one of the hosts is intercepted. You can configure the **overlap-tap enable** command to separately intercept the ASR 9000 ingress as well as egress traffic for both the mediation devices.
- Lawful intercept does not provide support for these features on Cisco ASR 9000 Series Router:
  - IPv6 multicast tapping
  - IPv4 multicast tapping
  - Per tap drop counter
  - IPv6 intercept on gigabit ethernet LCs
  - IPv6 MD encapsulation
  - Replicating a single tap to multiple MDs
  - Tapping of tag packets
  - Tapping L2 flows
  - RTP encapsulation
  - Encryption and integrity checking of replication device
  - GRE encapsulation
  - MPLS encapsulation

Note Per tap drop counter support is available only for ASR9000-SIP-700 line card, and not for ethernet line cards.

- Lawful intercept is applied only on ingress traffic.

  Traffic is intercepted, when it arrives as pure IP in the following scenarios:

  - For label imposition direction
  - When it arrives from the core after PHP action.

  Traffic is not intercepted in the following criteria:

  - When it arrives from the core as MPLS encapsulated (with VPN label) for the label disposition direction.
  - For GRE encapsulated packets.

- Lawful intercept traffic cannot be transmitted over the management interface.

Information About Lawful Intercept Implementation

Cisco lawful intercept is based on service-independent intercept (SII) architecture and SNMPv3 provisioning architecture. SNMPv3 addresses the requirements to authenticate data origin and ensure that the connection from the router to the MD is secure. This ensures that unauthorized parties cannot forge an intercept target.

Lawful intercept offers these capabilities:

- Voice-over IP (VoIP) and data session intercept provisioning from the MD using SNMPv3
- Delivery of intercepted VoIP and data session data to the MD
- SNMPv3 lawful intercept provisioning interface
- Lawful intercept MIB: CISCO-TAP2-MIB, version 2
- CISCO-IP-TAP-MIB manages the Cisco intercept feature for IP and is used along with CISCO-TAP2-MIB to intercept IP traffic.
- User datagram protocol (UDP) encapsulation to the MD
- Replication and forwarding of intercepted packets to the MD
- Voice-over IP (VoIP) call intercept, based on any rules configured for received packets.
- Voice-over IP (VoIP) intercept with LI-enabled call agent
- Data session call intercept based on IP address

Interception Mode

The lawful intercept has two interception modes:

- **Global LI:** The taps are installed on all the line cards in the ingress direction. With the global tap, the traffic for the target can be intercepted regardless of ingress point. Only the tap that has wild cards in the interface field is supported.
- **Interface LI:** Taps each packet that is entering or leaving an interface without any additional filters.

Overlapping Taps

Traffic interception can be configured for two inter-communicating intercepted hosts using overlapping taps.

For example, consider two taps, one configured for all traffic from source address A and another for all traffic going to destination address B. When a packet arrives with source address A and destination address B, the packet is tapped by TAP1 in ingress and TAP2 in egress, and copies will be generated and forwarded to both mediation devices. Overlapping taps can be enabled using **overlap-tap enable** command in Global configuration mode.

From Cisco IOS XR Software Release 7.0.1 and later, the Lawful Intercept overlapping tap functionality is available on Cisco ASR 9000 4th Generation QSFP28 based dense 100GE line cards (A9K-8X100GE-X-TR, A9K-16X100GE-TR, A9K-32X100GE-TR and A99-16x100-X-SE) as well.

Provisioning for VoIP Calls

Lawful Intercept provisioning for VoIP occurs in these ways:

- Security and authentication occurs because users define this through SNMPv3.
- The MD provisions lawful intercept information using SNMPv3.
- Network management occurs through standard MIBs.

Call Interception

VoIP calls are intercepted in this manner:

- The MD uses configuration commands to configure the intercept on the call control entity.
- The call control entity sends intercept-related information about the target to the MD.
- The MD initiates call content intercept requests to the content IAP router or trunk gateway through SNMPv3.
- The content IAP router or trunk gateway intercepts the call content, replicates it, and sends it to the MD in Packet Cable Electronic Surveillance UDP format. Specifically, the original packet starting at the first byte of the IP header is prefixed with a four-byte CCCID supplied by the MD in TAP2-MIB. It is then put into a UDP frame with the destination address and port of the MD.
- After replicated VoIP packets are sent to the MD, the MD then forwards a copy to a law-enforcement-agency-owned collection function, using a recognized standard.

Provisioning for Data Sessions

Provisioning for data sessions occurs in a similar way to the way it does for lawful intercept for VoIP calls. (See Provisioning for VoIP Calls, on page 305.)

Data Interception

Data are intercepted in this manner:

- If a lawful intercept-enabled authentication or accounting server is not available, a sniffer device can be used to detect the presence of the target in the network.
  - The MD uses configuration commands to configure the intercept on the sniffer.
  - The sniffer device sends intercept-related information about the target to the MD.
- The MD initiates communication content intercept requests to the content IAP router using SNMPv3.
- The content IAP router intercepts the communication content, replicates it, and sends it to the MD in UDP format.
- Intercepted data sessions are sent from the MD to the collection function of the law enforcement agency, using a supported delivery standard for lawful intercept.

Information About the MD

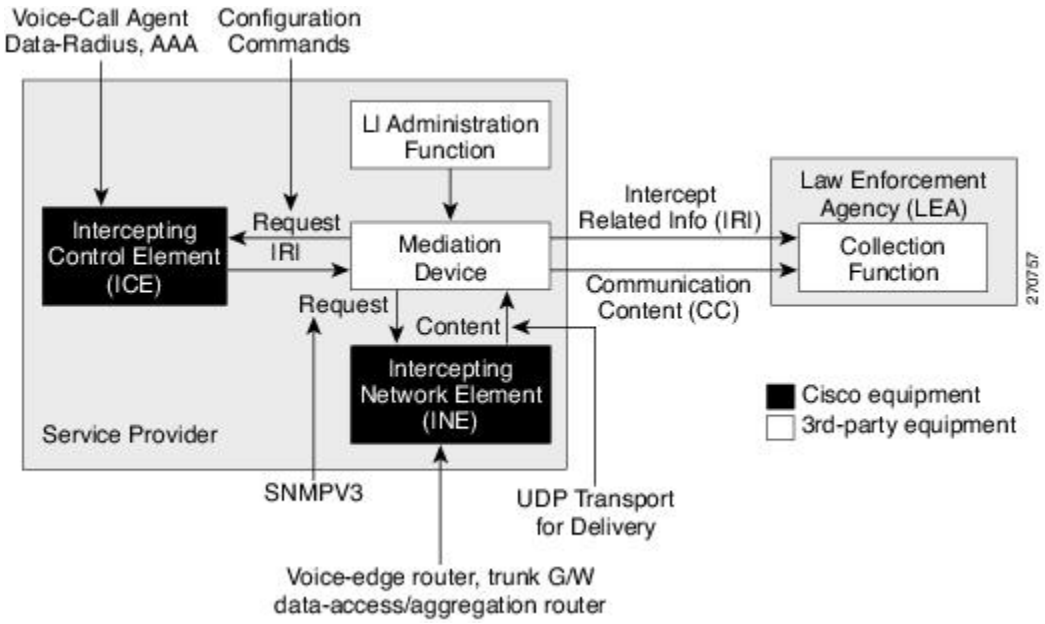
The MD performs these tasks:

- Activates the intercept at the authorized time and removes it when the authorized time period elapses.
- Periodically audits the elements in the network to ensure that:
  - *only* authorized intercepts are in place.
  - *all* authorized intercepts are in place.

Lawful Intercept Topology

This figure shows intercept access points and interfaces in a lawful intercept topology for both voice and data interception.

Figure 16: Lawful Intercept Topology for Both Voice and Data Interception



Layer 2 Lawful Intercept

You can configure SNMP-based lawful intercept on a layer 2 interface. This intercepts all traffic passing through the particular interface.

Lawful Intercept on MPLS Layer 3

Table 39: Feature History Table

Feature Name	Release Information	Feature Description
Lawful Intercept on MPLS Layer 3	Release 24.2.1	You can now enable lawful intercept (LI) consistently across different underlying transport technologies. This is made possible by enabling the lawful intercept at the MPLS Layer 3 in the network. You can enable lawful intercept using the SNMPv3. Additionally, Lawful Intercept enablement at MPLS Layer 3 allows managing LI centrally, which makes it easier for service providers to comply with Law Enforcement Agencies (LEAs) requests and reduce administrative overhead.

You can now configure lawful intercept at the MPLS Layer 3 in the network using SNMP. This intercepts all traffic passing through the particular interface including the traffic with VPN labels. This is made possible by intercepting the traffic based on the VRF (VPN Routing and Forwarding) filter of the classing tap.

To add TAP for an MPLS Layer 3 network, use the following command in the SNMP server:

```
setany -v3 <ip-address> <user> citapStreamInterface.1.1 0 citapStreamAddrType.1.1 <ipv4/ipv6>
 citapStreamSourceAddress.1.1 "<ip>" citapStreamSourceLength.1.1 <length> citapStreamVRF.1.1
 <vrf_name> citapStreamStatus.1.1 createAndGo
```

In the command:

- **ip-address** is the IP address of the SNMP agent (router).
- **user** is the username for SNMPv3 authentication.
- **ipv4/ipv6** is the address type of the SNMP agent (router).
- **ip** is the actual source IP address you want to monitor or capture.
- **vrf_name** is the VRF (Virtual Routing and Forwarding) instance that you want the LI to use.

Example:

```
setany -v3 192.0.2.5 li-user citapStreamInterface.1.1 0 citapStreamAddrType.1.1 ipv4
citapStreamSourceAddress.1.1 "203.0.113.17" citapStreamSourceLength.1.1 32 citapStreamVRF.1.1
 l3vpn citapStreamStatus.1.1 createAndGo
```

For more information on enabling MPLS Layer 3, refer to the *MPLS Layer 3 VPN Configuration Guide for Cisco ASR 9000 Series Routers*.

For more information on enabling SNMP-based lawful intercept, refer How to Configure SNMPv3 Access for Lawful Intercept, on page 317.

Scale or Performance Improvement

New enhancements introduced on the Cisco ASR 9000 Series Router in terms of scalability and performance for lawful intercept are:

- IPv4 lawful intercept tap limit is 1000 taps per IPv4 except for the A9K-8x100G-LB-SE and A9K-8x100G-LB-TR line cards. These line cards have a tap limit of 2000 taps per IPv4.
- IPv6 lawful intercept tap limit is 1000 taps per IPv6.
- Interception rate is:
  - 50 Mbps per network processor (NP) for ASR9000-SIP-700 line card.
  - 100 Mbps for Gigabit Ethernet line cards.
  - 500 Mbps for Modular Weapon-X line cards.
  - 1000 Mbps for 100GE line cards.
- Support upto 512 MDs.

Intercepting IPv4 and IPv6 Packets

This section provides details for intercepting IPv4 and IPv6 packets supported on the Cisco ASR 9000 Series Router.

Lawful Intercept Filters

The filters used for classifying a tap are:

- IP address type
- Destination address
- Destination mask
- Source address
- Source mask
- ToS (Type of Service) and ToS mask
- Protocol
- Destination port with range
- Source port with range
- VRF (VPN Routing and Forwarding)
- Flow ID

Intercepting Packets Based on Flow ID (Applies to IPv6 only)

To further extend filteration criteria for IPv6 packets, an additional support to intercept IPv6 packets based on flow ID has been introduced on the Cisco ASR 9000 Series Router. All IPv6 packets are intercepted based on the fields in the IPv6 header which comprises numerous fields defined in IPv6 Header Field Details table:

Note The field length or payload length is not used for intercepting packets.

Table 40: IPv6 Header Field Details

IPv6 Field Name	Field Description	Field Length
Version	IPv6 version number.	4 bits
Traffic Class	Internet traffic priority delivery value.	8 bits
Flow ID (Flow Label)	Used for specifying special router handling from source to destination(s) for a sequence of packets.	20 bits
Payload Length	Specifies the length of the data in the packet. When cleared to zero, the option is a hop-by-hop Jumbo payload.	16 bits unassigned
Next Header	Specifies the next encapsulated protocol. The values are compatible with those specified for the IPv4 protocol field.	8 bits

IPv6 Field Name	Field Description	Field Length
Hop Limit	For each router that forwards the packet, the hop limit is decremented by 1. When the hop limit field reaches zero, the packet is discarded. This replaces the TTL field in the IPv4 header that was originally intended to be used as a time based hop limit.	8 bits unsigned
Source Address	The IPv6 address of the sending node.	16 bytes
Destination Address	The IPv6 address of the destination node.	16 bytes

The flow ID or flow label is a 20 bit field in the IPv6 packet header that is used to discriminate traffic flows. Each flow has a unique flow ID. The filteration criteria to intercept packets matching a particular flow ID is defined in the tap configuration file. From the line card, the intercepted mapped flow IDs are sent to the next hop, specified in the MD configuration file. The intercepted packets are replicated and sent to the MD fom the line card.

Intercepting VRF (6VPE) and 6PE Packets

This section provides information about intercepting VRF aware packets and 6PE packets. Before describing how it works, a basic understanding of 6VPE networks is discussed.

The MPLS VPN model is a true peer VPN model. It enforces traffic separations by assigning unique VPN route forwarding (VRF) tables to each customer's VPN at the provider content IAP router. Thus, users in a specific VPN cannot view traffic outside their VPN.

Cisco ASR 9000 Series Router supports intercepting IPv6 packets of the specified VRF ID for 6VPE. To distiguish traffic on VPN, VRFs are defined containing a specific VRF ID. The filter criteria to tap a particular VRF ID is specified in the tap. IPv6 packets are intercepted with the VRF context on both scenarios: imposition (ip2mpls) and disposition (mpls2ip).

The 6PE packets carry IPv6 packets over VPN. The packets do not have a VRF ID. Only IP traffic is intercepted; no MPLS based intercepts are supported. The IPv6 traffic is intercepted at the content IAP of the MPLS cloud at imposition (ip2mpls) and at disposition (mpls2ip).

Intercepting IPv6 packets is also performed for ip2tag and tag2ip packets. Ip2tag packets are those which are converted from IPv6 to Tagging (IPv6 to MPLS), and tag2ip packects are those which are converted from Tagging to IPv6 (MPLS to IPv6) at the provider content IAP router.

Encapsulation Type Supported for Intercepted Packets

Intercepted packets mapping the tap are replicated, encapsulated, and then sent to the MD. IPv4 and IPv6 packets are encapsulated using UDP (User Datagram Protocol) encapsulation. The replicated packets are forwarded to MD using UDP as the content delivery protocol. Only IPv4 MD encapsulation is supported.

The intercepted packet gets a new UDP header and IPv4 header. Information for IPv4 header is derived from MD configuration. Apart from the IP and UDP headers, a 4 byte channel identifier (CCCID) is also inserted after the UDP header in the packet. After adding the MD encapsulation, if the packet size is above the MTU, the egress LC CPU fragments the packet. Moreover, there is a possibility that the packet tapped is already a fragment. Each tap is associated with only one MD. Cisco ASR 9000 Series Router does not support forwarding replicated packets to multiple MDs.

Note Encapsulation types, such as RTP and RTP-NOR, are not supported.

Per Tap Drop Counter Support

Cisco ASR 9000 Series Router line cards provide SNMP server as an interface to export each tap forwarded to MD packet and drop counts. Any intercepted packets that are dropped prior to getting forwarded to the MD due to policer action are counted and reported. The drops due to policer action are the only drops that are counted under per tap drop counters. If a lawful intercept filter is modified, the packet counts are reset to 0.

Note Per tap drop counter support is available only for ASR9000-SIP-700 line card, and not for ethernet line cards.

High Availability for Lawful Intercept

High availability for lawful intercept provides operational continuity of the TAP flows and provisioned MD tables to reduce loss of information due to route processor fail over (RPFO).

To achieve continuous interception of a stream, when RP fail over is detected; MDs are required to re-provision all the rows relating to CISCO-TAP2-MIB, CISCO-IP-TAP-MIB, and CISCO-USER-CONNECTION-TAP-MIB to synchronize database view across RP and MD.

Note The high availability for lawful intercept is enabled by default from Release 4.2.0 onwards.

Preserving TAP and MD Tables during RP Fail Over

At any point in time, MD has the responsibility to detect the loss of the taps via SNMP configuration process.

After RPFO is completed, MD should re-provision all the entries in the stream tables, MD tables, and IP taps with the same values they had before fail over. As long as an entry is re-provisioned in time, existing taps will continue to flow without any loss.

The following restrictions are listed for re-provisioning MD and tap tables with respect to behavior of SNMP operation on citapStreamEntry, cTap2StreamEntry, cTap2MediationEntry MIB objects:

- After RPFO, table rows that are not re-provisioned, shall return NO_SUCH_INSTANCE value as result of SNMP Get operation.
- Entire row in the table must be created in a single configuration step, with exactly same values as before RPFO, and with the rowStatus as CreateAndGo. Only exception is the cTap2MediationTimeout object, that should reflect valid future time.

Replay Timer

The replay timer is an internal timeout that provides enough time for MD to re-provision tap entries while maintaining existing tap flows. It resets and starts on the active RP when RPFO takes place. The replay timer is a factor of number of LI entries in router with a minimum value of 10 minutes.

After replay timeout, interception stops on taps that are not re-provisioned.

Note In case high availability is not required, MD waits for entries to age out after fail over. MD cannot change an entry before replay timer expiry. It can either reinstall taps as is, and then modify; or wait for it to age out.

Installing Lawful Intercept (LI) Package

As LI is not a part of the Cisco IOS XR image by default, you need to install it separately.

Installing and Activating the LI Package

The Package Installation Envelope (PIE) files, are installable software files with the .pie extension. PIE files are used to copy one or more software components onto the router. A PIE may contain a single component, a group of components (called a package), or a set of packages (called a composite package).

Use the **show install committed** command in EXEC mode to verify the committed software packages.

To install the Lawful Intercept (LI) package, you must install and activate the **asr9k-li-px.pie**

For more information about installing PIEs, refer to *Upgrading and Managing Cisco IOS XR Software section* of the *System Management Configuration Guide for Cisco ASR 9000 Series Routers*.

SUMMARY STEPS

1. **admin**
2. **install add** *tftp://<IP address of tftp server>/<location of pie on server>*
3. **install activate** *device:package*
4. **install commit**
5. **exit**
6. **show install committed**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**admin** **Example:** `RP/0/RSP0/CPU0:router# admin`	Enters administration EXEC mode.

	Command or Action	Purpose
Step 2	**install add** *tftp://<IP address of tftp server>/<location of pie on server>* **Example:** `RP/0/RSP0/CPU0:router(admin)# install add tftp://172.201.11.140/auto/tftp-users1/asr9k-li-px.pie`	Copies the contents of a package installation envelope (PIE) file to a storage device.
Step 3	**install activate** *device:package* **Example:** `RP/0/RSP0/CPU0:router(admin)# install activate disk0:asr9k-li-px.pie`	Activates the respective package and adds more functionality to the existing software.
Step 4	**install commit** **Example:** `RP/0/RSP0/CPU0:router(admin)# install commit`	Saves the active software set to be persistent across designated system controller (DSC) reloads.
Step 5	**exit** **Example:** `RP/0/RSP0/CPU0:router(admin)# exit`	Exits from the admin mode.
Step 6	**show install committed** **Example:** `RP/0/RSP0/CPU0:router# show install committed`	Shows the list of the committed software packages.

Deactivating the LI PIE

To uninstall the Lawful Intercept package, deactivate asr9k-li-px.pie as shown in the following steps:

Note You might experience interface or protocol flaps while uninstalling or deactivating the LI PIE. Hence, we recommend you to perform this activity during a maintenance window.

SUMMARY STEPS

1. **admin**
2. **install deactivate** *device:package*
3. **install commit**
4. **install remove** *device:package*
5. **exit**
6. **show install committed**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**admin** **Example:** `RP/0/RSP0/CPU0:router# admin`	Enters administration EXEC mode.
Step 2	**install deactivate** *device:package* **Example:** `RP/0/RSP0/CPU0:router(admin)# install deactivate disk0:asr9k-li-px.pie`	Activates the respective package and adds more functionality to the existing software.
Step 3	**install commit** **Example:** `RP/0/RSP0/CPU0:router(admin)# install commit`	Saves the active software set to be persistent across designated system controller (DSC) reloads.
Step 4	**install remove** *device:package* **Example:** `RP/0/RSP0/CPU0:router(admin)# install remove disk0:asr9k-li-px.pie`	Saves the active software set to be persistent across designated system controller (DSC) reloads.
Step 5	**exit** **Example:** `RP/0/RSP0/CPU0:router(admin)# exit`	Exits from the admin mode.
Step 6	**show install committed** **Example:** `RP/0/RSP0/CPU0:router# show install committed`	Shows the list of the committed software packages.

Upgrade and Downgrade Scenarios for the Lawful Intercept package

This section describes the possible upgrade and downgrade scenarios with respect to the Lawful Intercept (LI) package.

This example configuration demonstrates how to upgrade or downgrade the Cisco IOS XR software with or without the LI package. Suppose you have two versions of software images, V1 and V2. If you want to upgrade or downgrade from V1 to V2 without the LI package, you need to perform the following steps for the upgrade or the downgrade procedure:

Note Ensure that you use Turbo Boot to load the image for the downgrade process.

1. Ensure that the device has booted with the V1 image. Check the Package Installation Envelope (PIE) files that have been installed in V1.
2. Save all the PIE files that exist in V2 in the Trivial File Transfer Protocol (TFTP) server. Copy the contents of the PIE files from the TFTP server by using the **install add** command in the admin mode.

```
RP/0/RSP0/CPU0:router(admin)# install add tar
tftp://223.255.254.254/install/files/pies.tar
```

3. To activate all the PIE files in V2 at once, run the following commands based on the type of upgrade:

At any point during the upgrade or the downgrade process, you can check the progress by using the **show install request** or the **show issu** command.

Some of the conventions that are followed in describing these scenarios are:

- Release 4.3.1 base image: It is the Cisco IOS XR software for Release 4.3.1 that contains Cisco LI by default.
- Release 4.3.2 base image: It is the Cisco IOS XR software for Release 4.3.2 that does not contain Cisco LI by default.
- Separate LI package: It is the LI package that needs to be installed separately for Release 4.3.2 and higher versions.

Table 41: Upgrade Scenarios

Upgrade From	Upgrade To	Result	Supported
Release 4.3.1 base image	Release 4.3.2 base image	Before the upgrade, the LI has to be configured and provisioned completely. After the upgrade to Release 4.3.2 version without the LI package, you cannot configure or provision LI.	Yes
Release 4.3.1 base image	Release 4.3.2 base image with the separate LI package	The Upgrade will reload the router. After the upgrade process completes, you need to reconfigure LI MDs/TAPs from the SNMP server. Also, all the LI configurations made in the earlier version is accepted.	Yes
Release 4.3.2 base image with the separate LI package	Release 4.3.3 base image with the separate LI package	After the upgrade, the LI configuration is not retained.	Yes
Release 4.3.2 base image with the separate LI package	Release 4.3.3 base image without the separate LI package	This upgrade is not possible as the installation process will not proceed without the LI PIE.	No

Upgrade From	Upgrade To	Result	Supported
Release 4.3.2 base image without the separate LI package	Release 4.3.3 base image with the separate LI package	This upgrade is possible.	Yes
ISSU for Release 4.3.1 base image	Release 4.3.2 with the separate LI package	After this upgrade, to retain the LI configuration, you have to replay the configuration before the replay timeout occurs.	Yes
ISSU for Release 4.3.2 base image with the separate LI package	Release 4.3.3 with the separate LI package	After this upgrade, to retain the LI configuration, you have to replay the configuration before the replay timeout occurs.	Yes

Table 42: Downgrade Scenarios

Downgrade From	Downgrade To	Result	Supported
Release 4.3.2 base image without the separate LI package	Release 4.3.1 base image	After the downgrade, begin the provisioning process of LI.	Yes
Release 4.3.2 base image with the separate LI package	Release 4.3.1 base image	This scenario is not supported.	No
Release 4.3.3 base image with the separate LI package	Release 4.3.2 base image with the separate LI package	After the downgrade, the LI configuration is not retained. You have to provision the LI once again.	Yes
Release 4.3.3 base image with the separate LI package	Release 4.3.2 base image without the LI package	After the downgrade, the LI configuration is lost. You will not be able to provision it after downgrade.	Yes
Release 4.3.3 base image	Release 4.3.2 base image with the separate LI package	The LI configuration is accepted and can be provisioned only after the downgrade.	Yes
ISSU			No

How to Configure SNMPv3 Access for Lawful Intercept

Perform these procedures in the order presented to configure SNMPv3 for the purpose of Lawful Intercept enablement:

Disabling SNMP-based Lawful Intercept

Lawful Intercept is enabled by default on the Cisco ASR 9000 Series Router after installing and activating the **asr9k-li-px.pie**.

- To disable Lawful Intercept, enter the **lawful-intercept disable** command in global configuration mode.
- To re-enable it, use the **no** form of this command.

Disabling SNMP-based Lawful Intercept: Example

```
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# lawful-intercept disable
```

Note The **lawful-intercept disable** command is available on the router, only after installing and activating the **asr9k-li-px.pie**.

All SNMP-based taps are dropped when lawful intercept is disabled.

Configuring the Inband Management Plane Protection Feature

If MPP was not earlier configured to work with another protocol, then ensure that the MPP feature is also not configured to enable the SNMP server to communicate with the mediation device for lawful interception. In such cases, MPP must be configured specifically as an inband interface to allow SNMP commands to be accepted by the router, using a specified interface or all interfaces.

Note Ensure this task is performed, even if you have recently migrated to Cisco IOS XR Software from Cisco IOS, and you had MPP configured for a given protocol.

For lawful intercept, a loopback interface is often the choice for SNMP messages. If you choose this interface type, you must include it in your inband management configuration.

For a more detailed discussion of the inband management interface, see the Inband Management Interface, on page 328.

Related Tasks

- Configuring a Device for Management Plane Protection for an Inband Interface, on page 330

Related Examples

- Configuring the Inband Management Plane Protection Feature: Example, on page 322

Enabling the Mediation Device to Intercept VoIP and Data Sessions

The following SNMP server configuration tasks enable the Cisco SII feature on a router running Cisco IOS XR Software by allowing the MD to intercept VoIP or data sessions.

SUMMARY STEPS

1. **configure**
2. **snmp-server view** *view-name* **ciscoTap2MIB included**
3. **snmp-server view** *view-name* **ciscoUserConnectionTapMIB included**
4. **snmp-server group** *group-name* **v3auth read** *view-name* **write** *view-name* **notify** *view-name*
5. **snmp-server host** *ip-address* **traps version 3 auth** *username* **udp-port** *port-number*
6. **snmp-server user** *mduser-id groupname* **v3 auth md5** md-*password*
7. Use the **commit** or **end** command.
8. show snmp users
9. show snmp group
10. show snmp view

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**snmp-server view** *view-name* **ciscoTap2MIB included** **Example:** `RP/0//CPU0:router(config)# snmp-server view TapName ciscoTap2MIB included`	Creates or modifies a view record and includes the CISCO-TAP2-MIB family in the view. The SNMP management objects in the CISCO-TAP2-MIB that controls lawful intercepts are included. This MIB is used by the mediation device to configure and run lawful intercepts on targets sending traffic through the router.
Step 3	**snmp-server view** *view-name* **ciscoUserConnectionTapMIB included** **Example:** `RP/0//CPU0:router(config)# snmp-server view TapName ciscoUserConnectionTapMIB included`	Creates or modifies a view record and includes the CISCO-USER-CONNECTION-TAP-MIB family, to manage the Cisco intercept feature for user connections. This MIB is used along with the CISCO-TAP2-MIB to intercept and filter user traffic.

	Command or Action	Purpose
Step 4	**snmp-server group** *group-name* **v3auth read** *view-name* **write** *view-name* **notify** *view-name* **Example:** `RP/0//CPU0:router(config)# snmp-server group TapGroup v3 auth read TapView write TapView notify TapView`	Configures a new SNMP group that maps SNMP users to SNMP views. This group must have read, write, and notify privileges for the SNMP view.
Step 5	**snmp-server host** *ip-address* **traps version 3 auth** *username* **udp-port** *port-number* **Example:** `RP/0//CPU0:router(config)# snmp-server host 223.255.254.224 traps version 3 auth bgreen udp-port 2555`	Specifies SNMP trap notifications, the version of SNMP to use, the security level of the notifications, and the recipient (host) of the notifications.
Step 6	**snmp-server user** *mduser-id groupname* **v3 auth md5** md-*password* **Example:** `RP/0//CPU0:router(config)# snmp-server mduser-id TapGroup v3 auth md5 mdpassword`	Configures the MD user as part of an SNMP group, using the v3 security model and the HMAC MD5 algorithm, which you associate with the MD password. • The *mduser-id* and *mdpassword* must match that configured on MD. Alternatively, these values must match those in use on the router. • Passwords must be eight characters or longer to comply with SNMPv3 security minimums. • Minimum Lawful Intercept security level is auth; The noauth option will not work, as it indicates noAuthnoPriv security level. The Lawful Intercept security level must also match that of the MD. • Choices other than MD5 are available on the router, but the MD values must match. Most MDs default to or support only MD5.
Step 7	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 8	show snmp users **Example:**	Displays information about each SNMP username in the SNMP user table.

	Command or Action	Purpose
	`RP/0//CPU0:router# show snmp users`	
Step 9	show snmp group **Example:** `RP/0//CPU0:router# show snmp group`	Displays information about each SNMP group on the network.
Step 10	show snmp view **Example:** `RP/0//CPU0:router# show snmp view`	Displays information about the configured views, including the associated MIB view family name, storage type, and status.

Adding MD and TAP Objects

To keep the MD row in active state, the following objects are mandatory:

- cTap2MediationDestAddressType
- cTap2MediationDestAddress
- cTap2MediationDestPort
- cTap2MediationSrcInteraface
- cTap2MediationTimeout
- cTap2MediationTransport
- cTap2MediationStatus

SUMMARY STEPS

1. Add MD.
2. Add TAP.
3. Activate TAP.

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	Add MD.
	Example:
	`setany -v3 <ip-address> <user>`
	`cTap2MediationDestAddressType.1 <ipv4/ipv6>`
	`cTap2MediationDestAddress.1 <"ip">`
	`cTap2MediationDestPort.1 "1234"`
	`cTap2MediationSrcInterface.1 0`
		Creates an MD for mediation services.
		To delete a MD, run:
		`setany -v3 <ip-address> <user>`
		`cTap2MediationStatus.1 6`
		`cTap2MediationStatus.1 = destroy(6)`

	Command or Action	Purpose
	cTap2MediationTransport.1 udp cTap2MediationNotificationEnable.1 true cTap2MediationTimeout.1 '7 de 6 14 3 4 5 6 2d 1 2' cTap2MediationStatus.1 createAndGo cTap2MediationDestAddressType.1 = ipv4(1) cTap2MediationDestAddress.1 = 46 01 01 02 cTap2MediationDestPort.1 = 1234 cTap2MediationSrcInterface.1 = 0 cTap2MediationTransport.1 = udp(1) cTap2MediationNotificationEnable.1 = true(1) cTap2MediationTimeout.1 = 2014-Jun-20,03:04:05.6,-1:2 cTap2MediationStatus.1 = createAndGo(4)	
Step 2	Add TAP. **Example:** setany -v3 <ip-address> <user> citapStreamInterface.1.1 0 citapStreamAddrType.1.1 <ipv4/ipv6> citapStreamSourceAddress.1.1 "5a 1 1 2" citapStreamSourceLength.1.1 32 citapStreamStatus.1.1 citapStreamInterface.1.1 = 0 citapStreamAddrType.1.1 = ipv4(1) citapStreamSourceAddress.1.1 = 5a 01 01 02 citapStreamSourceLength.1.1 = 32 citapStreamStatus.1.1 = createAndGo(4)	Creates a TAP for stream operation. To delete a TAP, run: setany -v3 <ip-address> <user> citapStreamStatus.1.1 6 cTap2StreamStatus.1.1 6 citapStreamStatus.1.1 = destroy(6) cTap2StreamStatus.1.1 = destroy(6)
Step 3	Activate TAP. **Example:** setany -v3 <ip-address> <user> cTap2StreamType.1.1 ip cTap2StreamInterceptEnable.1.1 true cTap2StreamStatus.1.1 createAndGo cTap2StreamType.1.1 = ip(1) cTap2StreamInterceptEnable.1.1 = true(1) cTap2StreamStatus.1.1 = createAndGo(4) **Example:** To add TAP for L2VPN networks setany -v3 <ip-address> <user> citapStreamInterface.4.1200 1125 citapStreamStatus.4.1200 createAndGo	Activates the TAP for stream operation.

Configuration Example for Inband Management Plane Feature Enablement

This example illustrates how to enable the MPP feature, which is disabled by default, for the purpose of lawful intercept.

Configuring the Inband Management Plane Protection Feature: Example

You must specifically enable management activities, either globally or on a per-inband-port basis, using this procedure. To globally enable inbound MPP, use the keyword **all** with the **interface** command, rather than use a particular interface type and instance ID with it.

```
RP/0//CPU0:router# configure
RP/0//CPU0:router(config)# control-plane
RP/0//CPU0:router(config-ctrl)# management-plane
RP/0//CPU0:router(config-mpp)# inband
RP/0//CPU0:router(config-mpp-inband)# interface loopback0
RP/0//CPU0:router(config-mpp-inband-Loopback0)# allow snmp
RP/0//CPU0:router(config-mpp-inband-Loopback0)# commit
RP/0//CPU0:router(config-mpp-inband-Loopback0)# exit
RP/0//CPU0:router(config-mpp-inband)# exit
RP/0//CPU0:router(config-mpp)# exit
RP/0//CPU0:router(config-ctr)# exit
RP/0//CPU0:router(config)# exit
RP/0//CPU0:router# show mgmt-plane inband interface loopback0

Management Plane Protection - inband interface


interface - Loopback0
        snmp configured -
                All peers allowed
RP/0//CPU0:router(config)# commit
```

Increased Lawful Interception rates

An increased Lawful Interception (LI) rate is a performance enhancement feature that

- enables higher-speed data interception for addressing legal compliance requirements, and
- supports a configurable lawful interception rate from 1 Gbps to 25 Gbps per NPU on a line card.

Table 43: Feature History Table

Feature Name	Release Information	Feature Description
Increased Lawful Interception rates	Release 25.3.1	You can now configure lawful interception (LI) rates up to 25 Gbps, allowing support for higher interception bandwidth requirements. Previously, the interception rate was limited to 1 Gbps. The feature introduces these changes: **CLI:** • **hardware profile li-recycle-bw**

Guidelines for configuring increased Lawful Interception rates

Platform support

This feature is available on ASR9000 platforms and 5th generation LSP line cards.

Interception rate limit

The configured interception rate applies per NPU on a line card.

For example, if a line card contains two NPUs and the rate limit is set to 25 Gbps, each NPU has a 25 Gbps limit.

Configure increased Lawful Interception rates

Use this procedure to configure increased Lawful Interception (LI) rates upto 25 Gbps.

Procedure

Step 1 Configure the LI rate.

Example:

```
Router# configure
Router(config)# hardware profile li-recycle-bw 25
Router(config)# commit
```

You can choose an LI rate from a range of 1 Gbps to 25 Gbps. The configured interception rate is applicable per NPU on a line card.

Step 2 Verify the configured LI rate is applied to the router.

Example:

```
Router#show running-config | in li-recycle-bw
hardware profile li-recycle-bw 25
```

Additional References

These sections provide references related to implementing lawful intercept.

Related Documents

Related Topic	Document Title
Lawful Intercept commands	*System Security Command Reference for Cisco ASR 9000 Series Routers*
Implementing SNMP	*System Management Configuration Guide for Cisco ASR 9000 Series Routers*
SNMP Server commands	*System Management Command Reference for Cisco ASR 9000 Series Routers*

Standards

Standards	Title
A modular, open architecture designed for simple implementation that easily interacts with third-party equipment to meet service provider lawful intercept requirements.	See RFC-3924 under RFCs, on page 324.
An application layer protocol that facilitates the exchange of management information between network devices. Part of the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite.	Simple Network Management Protocol Version 3 (SNMPv3)

MIBs

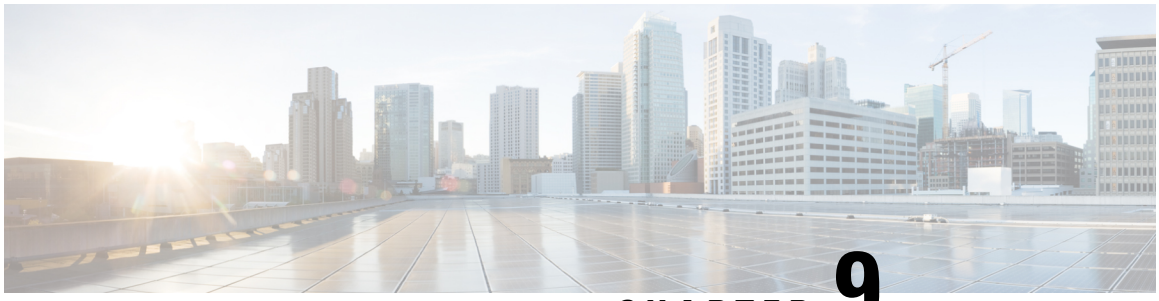
MIBs	MIBs Link
• CISCO-TAP2-MIB, version 2 • CISCO-IP-TAP-MIB	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC-3924	Cisco Architecture for Lawful Intercept in IP Networks

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access more content.	http://www.cisco.com/techsupport



CHAPTER 9

Implementing Management Plane Protection

The Management Plane Protection (MPP) feature in Cisco IOS XR software provides the capability to restrict the interfaces on which network management packets are allowed to enter a device. The MPP feature allows a network operator to designate one or more router interfaces as management interfaces.

Device management traffic may enter a device only through these management interfaces. After MPP is enabled, no interfaces except designated management interfaces accept network management traffic destined to the device. Restricting management packets to designated interfaces provides greater control over management of a device, providing more security for that device.

This module describes how to implement management plane protection on Cisco ASR 9000 Series Routers.

For information on MPP commands, see the *Management Plane Protection Commands* module in *System Security Command Reference for Cisco ASR 9000 Series Routers* .

Feature History for Implementing Management Plane Protection

Release	Modification
Release 3.7.2	This feature was introduced.

- Prerequisites for Implementing Management Plane Protection, on page 327
- Restrictions for Implementing Management Plane Protection, on page 327
- Information About Implementing Management Plane Protection, on page 328
- How to Configure a Device for Management Plane Protection, on page 330
- Configuration Examples for Implementing Management Plane Protection, on page 335
- Additional References, on page 337

Prerequisites for Implementing Management Plane Protection

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Restrictions for Implementing Management Plane Protection

The following restrictions are listed for implementing Management Plane Protection (MPP):

- Currently, MPP does not keep track of the denied or dropped protocol requests.
- MPP configuration does not enable the protocol services. MPP is responsible only for making the services available on different interfaces. The protocols are enabled explicitly.
- Management requests that are received on inband interfaces are not necessarily acknowledged there.
- Both Route Processor (RP) and distributed route processor (DRP) Ethernet interfaces are by default out-of-band interfaces and can be configured under MPP.
- The changes made for the MPP configuration do not affect the active sessions that are established before the changes.
- Currently, MPP controls only the incoming management requests for protocols, such as TFTP, Telnet, Simple Network Management Protocol (SNMP), Secure Shell (SSH), and HTTP.
- MPP does not support MIB.
- In a MPLS L3VPN, when MPP has VRF interface attached, it applies the VRF filter on an incoming interface through LPTS. When an incoming packet from the core interface has a different VRF, then MPP does not allow it.



Note When configuring a device for MPP for an inband interface the **Interface all** configuration does not apply specific VRF filter and allows traffic for all source and destination interfaces.

Information About Implementing Management Plane Protection

Before you enable the Management Plane Protection feature, you should understand the following concepts:

Inband Management Interface

An *inband management interface* is a Cisco IOS XR software physical or logical interface that processes management packets, as well as data-forwarding packets. An inband management interface is also called a *shared management interface*.

Out-of-Band Management Interface

Out-of-band refers to an interface that allows only management protocol traffic to be forwarded or processed. An *out-of-band management interface* is defined by the network operator to specifically receive network management traffic. The advantage is that forwarding (or customer) traffic cannot interfere with the management of the router, which significantly reduces the possibility of denial-of-service attacks.

Out-of-band interfaces forward traffic only between out-of-band interfaces or terminate management packets that are destined to the router. In addition, the out-of-band interfaces can participate in dynamic routing protocols. The service provider connects to the router's out-of-band interfaces and builds an independent overlay management network, with all the routing and policy tools that the router can provide.

Peer-Filtering on Interfaces

The peer-filtering option allows management traffic from specific peers, or a range of peers, to be configured.

Control Plane Protection Overview

A *control plane* is a collection of processes that run at the process level on a route processor and collectively provide high-level control for most Cisco IOS XR software functions. All traffic directly or indirectly destined to a router is handled by the control plane. Management Plane Protection operates within the Control Plane Infrastructure.

Management Plane

The *management plane* is the logical path of all traffic that is related to the management of a routing platform. One of three planes in a communication architecture that is structured in layers and planes, the management plane performs management functions for a network and coordinates functions among all the planes (management, control, and data). In addition, the management plane is used to manage a device through its connection to the network.

Examples of protocols processed in the management plane are Simple Network Management Protocol (SNMP), Telnet, HTTP, Secure HTTP (HTTPS), and SSH. These management protocols are used for monitoring and for command-line interface (CLI) access. Restricting access to devices to internal sources (trusted networks) is critical.

Management Plane Protection Feature

The MPP protection feature, as well as all the management protocols under MPP, are disabled by default. When you configure an interface as either out-of-band or inband, it automatically enables MPP. Consequently, this enablement extends to all the protocols under MPP.

If MPP is disabled and a protocol is activated, all interfaces can pass traffic.

When MPP is enabled with an activated protocol, the only default management interfaces allowing management traffic are the route processor (RP) and standby route processor (SRP) Ethernet interfaces. You must manually configure any other interface for which you want to enable MPP as a management interface, using the MPP CLI that follows. Afterwards, only the default management interfaces and those you have previously configured as MPP interfaces will accept network management packets destined for the device. All other interfaces drop such packets.



Note Logical interfaces (or any other interfaces not present on the data plane) filter packets based on the ingress physical interface.

After configuration, you can modify or delete a management interface.

Following are the management protocols that the MPP feature supports. These management protocols are also the only protocols affected when MPP is enabled.

- SSH, v1 and v2
- SNMP, all versions

- Telnet
- TFTP
- HTTP
- HTTPS

Benefits of the Management Plane Protection Feature

Implementing the MPP feature provides the following benefits:

- Greater access control for managing a device than allowing management protocols on all interfaces.
- Improved performance for data packets on non-management interfaces.
- Support for network scalability.
- Simplifies the task of using per-interface access control lists (ACLs) to restrict management access to the device.
- Fewer ACLs are needed to restrict access to the device.
- Prevention of packet floods on switching and routing interfaces from reaching the CPU.

How to Configure a Device for Management Plane Protection

This section contains the following tasks:

Configuring a Device for Management Plane Protection for an Inband Interface

Perform this task to configure a device that you have just added to your network or a device already operating in your network. This task shows how to configure MPP as an inband interface in which Telnet is allowed to access the router only through a specific interface.

Perform the following additional tasks to configure an inband MPP interface in non-default VRF.

- Configure the interface under the non-default inband VRF.
- Configure the global inband VRF.
- In the case of Telnet, configure the Telnet VRF server for the inband VRF.

SUMMARY STEPS

1. **configure**
2. control-plane
3. management-plane
4. inband
5. **interface** {*type instance* | **all**}
6. **allow** {*protocol* | **all**} [**peer**]
7. **address ipv4** {*peer-ip-address* | *peer ip-address/length*}

8. Use the **commit** or **end** command.
9. **show mgmt-plane** [**inband** | **out-of-band**] [**interface** {*type instance*}]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	control-plane
	Example:
	`RP/0/RSP0/CPU0:router(config)# control-plane`
	`RP/0/RSP0/CPU0:router(config-ctrl)#`
		Enters control plane configuration mode.

Step 3	management-plane
	Example:
	`RP/0/RSP0/CPU0:router(config-ctrl)#`
	`management-plane`
	`RP/0/RSP0/CPU0:router(config-mpp)#`
		Configures management plane protection to allow and disallow protocols and enters management plane protection configuration mode.

Step 4	inband
	Example:
	`RP/0/RSP0/CPU0:router(config-mpp)# inband`
	`RP/0/RSP0/CPU0:router(config-mpp-inband)#`
		Configures an inband interface and enters management plane protection inband configuration mode.

Step 5	**interface** {*type instance* | **all**}
	Example:
	`RP/0/RSP0/CPU0:router(config-mpp-inband)# interface GigabitEthernet 0/6/0/1`
	`RP/0/RSP0/CPU0:router(config-mpp-inband-Gi0_6_0_1)#`
		Configures a specific inband interface, or all inband interfaces. Use the **interface** command to enter management plane protection inband interface configuration mode.
		• Use the **all** keyword to configure all interfaces.

Step 6	**allow** {*protocol* | **all**} [**peer**] **Example:**	Configures an interface as an inband interface for a specified protocol or all protocols.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-mpp-inband-Gi0_6_0_1)# allow Telnet peer` `RP/0/RSP0/CPU0:router(config-telnet-peer)#`	• Use the *protocol* argument to allow management protocols on the designated management interface. • HTTP or HTTPS • SNMP (also versions) • Secure Shell (v1 and v2) • TFTP • Telnet • Use the **all** keyword to configure the interface to allow all the management traffic that is specified in the list of protocols. • (Optional) Use the **peer** keyword to configure the peer address on the interface.
Step 7	**address ipv4** {*peer-ip-address* | *peer ip-address/length*} **Example:** `RP/0/RSP0/CPU0:router(config-telnet-peer)# address ipv4 10.1.0.0/16`	Configures the peer IPv4 address in which management traffic is allowed on the interface. • Use the *peer-ip-address* argument to configure the peer IPv4 address in which management traffic is allowed on the interface. • Use the *peer ip-address/length* argument to configure the prefix of the peer IPv4 address.
Step 8	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 9	**show mgmt-plane** [**inband** | **out-of-band**] [**interface** {*type instance*}] **Example:** `RP/0/RSP0/CPU0:router# show mgmt-plane inband interface GigabitEthernet 0/6/0/1`	Displays information about the management plane, such as type of interface and protocols enabled on the interface. • (Optional) Use the **inband** keyword to display the inband management interface configurations that are the interfaces that process management packets as well as data-forwarding packets. • (Optional) Use the **out-of-band** keyword to display the out-of-band interface configurations.

	Command or Action	Purpose
		• (Optional) Use the **interface** keyword to display the details for a specific interface.

Configuring a Device for Management Plane Protection for an Out-of-band Interface

Perform the following tasks to configure an out-of-band MPP interface.

- Configure the interface under the out-of-band VRF.
- Configure the global out-of-band VRF.
- In the case of Telnet, configure the Telnet VRF server for the out-of-band VRF.

SUMMARY STEPS

1. **configure**
2. control-plane
3. management-plane
4. out-of-band
5. **vrf** *vrf-name*
6. **interface** {*type instance* | **all**}
7. **allow** {*protocol* | **all**} [**peer**]
8. **address ipv6** {*peer-ip-address* | *peer ip-address/length*}
9. Use the **commit** or **end** command.
10. **show mgmt-plane** [**inband** | **out-of-band**] [**interface** {*type instance*} | **vrf**]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	control-plane
	Example:
	`RP/0/RSP0/CPU0:router(config)# control-plane`
	`RP/0/RSP0/CPU0:router(config-ctrl)#`
		Enters control plane configuration mode.

	Command or Action	Purpose

Step 3	management-plane
	Example:
	`RP/0/RSP0/CPU0:router(config-ctrl)# management-plane`
	`RP/0/RSP0/CPU0:router(config-mpp)#`
		Configures management plane protection to allow and disallow protocols and enters management plane protection configuration mode.

Step 4	out-of-band
	Example:
	`RP/0/RSP0/CPU0:router(config-mpp)# out-of-band`
	`RP/0/RSP0/CPU0:router(config-mpp-outband)#`
		Configures out-of-band interfaces or protocols and enters management plane protection out-of-band configuration mode.

Step 5	**vrf** *vrf-name* **Example:** `RP/0/RSP0/CPU0:router(config-mpp-outband)# vrf target`	Configures a Virtual Private Network (VPN) routing and forwarding (VRF) reference of an out-of-band interface. • Use the *vrf-name* argument to assign a name to a VRF.

Step 6	**interface** {*type instance* | **all**}
	Example:
	`RP/0/RSP0/CPU0:router(config-mpp-outband)# interface GigabitEthernet 0/6/0/2`
	`RP/0/RSP0/CPU0:router(config-mpp-outband-Gi0_6_0_2)#`
		Configures a specific out-of-band interface, or all out-of-band interfaces, as an out-of-band interface. Use the **interface** command to enter management plane protection out-of-band configuration mode.
		• Use the **all** keyword to configure all interfaces.

Step 7	**allow** {*protocol* | **all**} [**peer**]
	Example:
	`RP/0/RSP0/CPU0:router(config-mpp-outband-Gi0_6_0_2)# allow TFTP peer`
	`RP/0/RSP0/CPU0:router(config-tftp-peer)#`
		Configures an interface as an out-of-band interface for a specified protocol or all protocols.
		• Use the *protocol* argument to allow management protocols on the designated management interface.
		  • HTTP or HTTPS
		  • SNMP (also versions)
		  • Secure Shell (v1 and v2)
		  • TFTP
		  • Telnet
		• Use the **all** keyword to configure the interface to allow all the management traffic that is specified in the list of protocols.

	Command or Action	Purpose
		• (Optional) Use the **peer** keyword to configure the peer address on the interface.
Step 8	**address ipv6** {*peer-ip-address* | *peer ip-address/length*} **Example:** `RP/0/RSP0/CPU0:router(config-tftp-peer)# address ipv6 33::33`	Configures the peer IPv6 address in which management traffic is allowed on the interface. • Use the *peer-ip-address* argument to configure the peer IPv6 address in which management traffic is allowed on the interface. • Use the *peer ip-address/length* argument to configure the prefix of the peer IPv6 address.
Step 9	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 10	**show mgmt-plane** [**inband** | **out-of-band**] [**interface** {*type instance*} | **vrf**] **Example:** `RP/0/RSP0/CPU0:router# show mgmt-plane out-of-band interface GigabitEthernet 0/6/0/2`	Displays information about the management plane, such as type of interface and protocols enabled on the interface. • (Optional) Use the **inband** keyword to display the inband management interface configurations that are the interfaces that process management packets as well as data-forwarding packets. • (Optional) Use the **out-of-band** keyword to display the out-of-band interface configurations. • (Optional) Use the **interface** keyword to display the details for a specific interface. • (Optional) Use the **vrf** keyword to display the Virtual Private Network (VPN) routing and forwarding reference of an out-of-band interface.

Configuration Examples for Implementing Management Plane Protection

This section provides the following configuration example:

Configuring Management Plane Protection: Example

The following example shows how to configure inband and out-of-band interfaces for a specific IP address under MPP:

```
configure
 control-plane
  management-plane
   inband
    interface all
     allow SSH
     !
    interface GigabitEthernet 0/6/0/0
     allow all
     allow SSH
     allow Telnet peer
      address ipv4 10.1.0.0/16
    !
   !
    interface GigabitEthernet 0/6/0/1
     allow Telnet peer
      address ipv4 10.1.0.0/16
     !
    !
   !
   out-of-band
    vrf my_out_of_band
    interface GigabitEthernet 0/6/0/2
     allow TFTP peer
      address ipv6 33::33
     !
    !
   !
  !
 !

show mgmt-plane

Management Plane Protection

inband interfaces
----------------------

interface - GigabitEthernet0_6_0_0
        ssh configured -
                All peers allowed
        telnet configured -
                peer v4 allowed - 10.1.0.0/16
        all configured -
                All peers allowed
interface - GigabitEthernet0_6_0_1
        telnet configured -
                peer v4 allowed - 10.1.0.0/16

interface - all
        all configured -
                All peers allowed

outband interfaces
----------------------
interface - POS0_6_0_2
```

```
        tftp configured -
                peer v6 allowed - 33::33

show mgmt-plane out-of-band vrf

Management Plane Protection -
        out-of-band VRF - my_out_of_band
```

Additional References

The following sections provide references related to implementing management plane protection.

Related Documents

Related Topic	Document Title
MPP commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Management Plane Protection Commands on System Monitoring Command Reference for Cisco ASR 9000 Series Routers*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

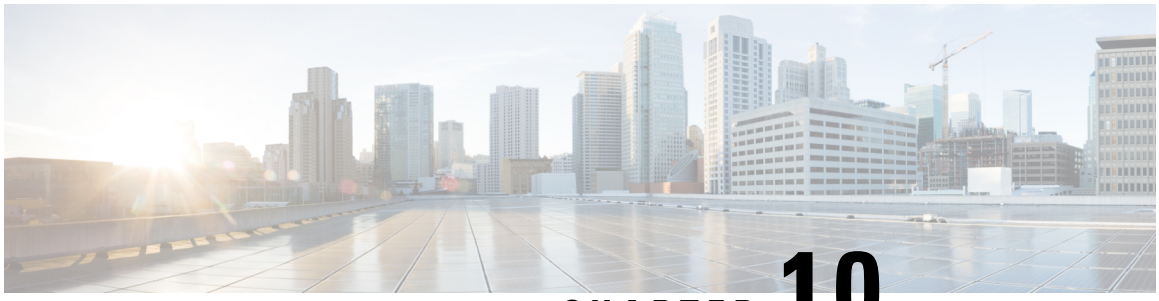
MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature.	—

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport



CHAPTER 10

Traffic Protection for Third-Party Applications

Traffic Protection for Third-Party Applications provides a mechanism for securing management traffic on the router. Without Traffic Protection for Third-Party Applications, if the service is enabled, the Cisco IOS XR allows the service traffic to pass through any interface with a network address.

Note Prior to Cisco IOS XR Release 6.5.2, Traffic Protection for Third-Party Applications was termed as MPP for Third-Party Applications.

Traffic Protection for Third-Party Applications helps in rate limiting or throttling the traffic through configuration with the help of LPTS. Traffic Protection for Third-Party Applications filters traffic based on the following tuples: address family, vrf, port, interface, local address and remote address.

- gRPC Protocol, on page 339
- Limitations for Traffic Protection for Third-Party Applications, on page 340
- Prerequisites for Traffic Protection for Third-Party Applications Over GRPC, on page 340
- Configuring Traffic Protection for Third-Party Applications, on page 340
- Troubleshooting Traffic Protection for Third-Party Applications, on page 341

gRPC Protocol

Google-defined Remote Procedure Calls (gRPC) is an open-source RPC framework. It is based on Protocol Buffers (Protobuf), which is an open source binary serialization protocol. gRPC provides a flexible, efficient, automated mechanism for serializing structured data, like XML, but is smaller and simpler to use. The user needs to define the structure by defining protocol buffer message types in .proto files. Each protocol buffer message is a small logical record of information, containing a series of name-value pairs.

Cisco gRPC Interface Definition Language (IDL) uses a set of supported RPCs such as get-config, merge-config, replace-config, cli-config, delete-config, cli-show, get-models, action-json, commit, and commit-replace. gRPC server runs in Extensible Manageability Services Daemon (emsd) process. gRPC client can be on any machine.

gRPC encodes requests and responses in binary. gRPC is extensible to other content types along with Protobuf. The Protobuf binary data object in gRPC is transported over HTTP/2.

Note It is recommended to configure TLS before enabling gRPC. Enabling gRPC protocol uses the default HTTP/2 transport with no TLS enabled on TCP. gRPC mandates AAA authentication and authorization for all gRPC requests. If TLS is not configured, the authentication credentials are transferred over the network unencrypted. Non-TLS mode can only be used in secure internal network.

gRPC supports distributed applications and services between a client and server. gRPC provides the infrastructure to build a device management service to exchange configuration and operational data between a client and a server. The structure of the data is defined by YANG models.

Limitations for Traffic Protection for Third-Party Applications

The following limitations are applicable for the Traffic Protection for Third-Party Applications:

- If the TPA entry is configured with only the active RP management interface and redundancy switchover is performed, the gRPC connection fails.

Prerequisites for Traffic Protection for Third-Party Applications Over GRPC

Ensure that the gRPC is configured.

gRPC Configuration

```
Router(config)# grpc port port-number
Router(config)# grpc no-tls
Router(config-grpc)# commit
```

Running Configuration

```
Router# show running-config grpc

grpc port 57600
 no-tls
!
```

Configuring Traffic Protection for Third-Party Applications

The following task shows how to configure traffic protection for third-party applications

```
RP/0/0/CPU0:ios#configure
RP/0/0/CPU0:ios(config)#tpa
RP/0/0/CPU0:ios(config-tpa)#vrf default
RP/0/0/CPU0:ios(config-tpa-vrf)#address-family ipv4
RP/0/0/CPU0:ios(config-tpa-vrf-afi)#protection
RP/0/0/CPU0:ios(config-tpa-vrf-afi-prot)#allow protocol tcp local-port port-number
remote-address IP remote address interface interface-name local-address IP local address
```

Note Do not add loopback interface in configuration. Adding a loopback interface in configuration leads to misconfiguration. This is because the packets do not arrive on loopback interface as the packets reaching the loopback will be blocked.

Running Configuration

```
Router# show running-config
tpa
vrf default
address-family ipv4
protection
allow protocol tcp local-port 57600 remote-address 10.0.0.2/32 local-address 192.168.0.1/32
allow protocol tcp local-port 57600 remote-address 10.0.1.0/24 local-address 192.168.0.1/32
allow protocol tcp local-port 57600 remote-address 10.0.2.0/24 local-address 192.168.0.1/32
address-family ipv6
protection
allow protocol tcp local-port 57600 remote-address 2001:DB8::1/128 local-address
2001:DB8:0:ABCD::1/128
allow protocol tcp local-port 57600 remote-address 2001:DB8::2/128 local-address
2001:DB8:0:ABCD::1/128
allow protocol tcp local-port 57600 remote-address 2001:DB8::3/128 local-address
2001:DB8:0:ABCD::1/128
!
   !
  !
!
```

Troubleshooting Traffic Protection for Third-Party Applications

The following show command output verifies whether Third-Party Applications is configured or not.

```
Router# show running-config grpc

grpc
 no-tls
!
```

The following show command output displays the TPA configuration.

```
Router# show running-config tpa


  tpa
   vrf default
    address-family ipv4
    allow local-port 57600 protocol tcp inter mgmtEth 0/RP0/CPU0/0 local-address
192.168.0.1/32 remote-address 10.0.0.2/32
    !
```

gRPC Configuration without TPA

```
Router# show kim lpts database

State:
 Prog - Programmed in hardware
 Cfg  - Configured, not yet programmed
 Ovr  - Not programmed, overridden by user configuration
 Intf - Not programmed, interface does not exist
```

```
Owner  AF Proto State       Interface      VRF          Local ip,port > Remote ip,port
------ -- ----- ----- -------------- ------------ ------------------------------
Linux   2     6          Prog                          global-vrf             any,57600
 > any,0

Router# show lpts bindings brief | include TPA
 0/RP0/CPU0 TPA  LR IPV4 TCP    default   any          any,57600 any
```

gRPC Configuration with TPA

The following show command output displays the things that are configured in the LPTS database. It also checks if gRPC configuration is owned by Linux without using any filters.

```
Router# show kim lpts database

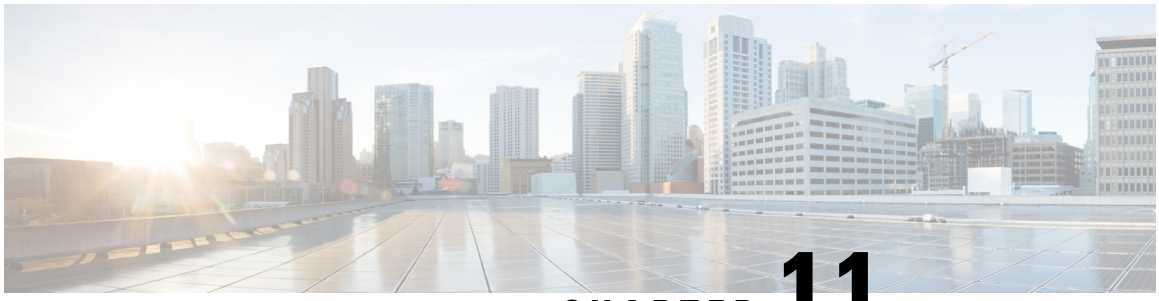
State:
 Prog - Programmed in hardware
 Cfg  - Configured, not yet programmed
 Ovr  - Not programmed, overridden by user configuration
 Intf - Not programmed, interface does not exist

Owner  AF Proto State Interface      VRF          Local ip,port > Remote ip,port
------ -- ----- ----- -------------- ------------ ------------------------------
Client  2     6 Prog                 default      192.168.0.1/32,57600  > 10.0.0.2/32,0
Linux   2     6 Ovr                  global-vrf   any,57600  > any,0

Router# show lpts bindings brief | include TPA

 0/RP0/CPU0 TPA  LR IPV4 TCP    default   Mg0/RP0/CPU0/0 192.168.0.1,57600 10.0.0.2
Router#
Router# 0/RP0/ADMIN0:Mar 19 15:22:26.837 IST: pm[2433]:
%INFRA-Process_Manager-3-PROCESS_RESTART : Process tams (IID: 0) restarted
```

For more information on **tpa** and **allow local-port** commands, see *System Security Command Reference for Cisco ASR 9000 Series Routers*.



CHAPTER 11

Configuring Software Authentication Manager

Software Authentication Manager (SAM) is a component of the the Cisco ASR 9000 Series Router operating system that ensures that software being installed on the router is safe, and that the software does not run if its integrity has been compromised.

For information on SAM commands, see the *Software Authentication Manager Commands* module in *System Security Command Reference for Cisco ASR 9000 Series Routers*.

For information on setting the system clock, see the **clock set** command in *Clock Commands* module in *System Management Command Reference for Cisco ASR 9000 Series Routers*.

Feature History for Configuring Software Authentication Manager

Release	Modification
Release 3.7.2	This feature was introduced.

- Prerequisites for Configuring Software Authentication Manager, on page 343
- Information about Software Authentication Manager, on page 343
- How to set up a Prompt Interval for the Software Authentication Manager, on page 344

Prerequisites for Configuring Software Authentication Manager

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Information about Software Authentication Manager

For SAM to verify software during installation, the software to be installed must be in a Packager for IOS/ENA (PIE) format. PIEs are digitally signed and SAM verifies the digital signature before allowing bits from that PIE to reside on the router. Each time an installed piece of software is run, SAM ensures that the integrity of the software is not been compromised since it was installed. SAM also verifies that software preinstalled on a flash card has not been tampered with while in transit.

When the initial image or a software package update is loaded on the router, SAM verifies the validity of the image by checking the expiration date of the certificate used to sign the image. If an error message is displayed

indicating that your certificate has expired, check the system clock and verify that it is accurate. If the system clock is not set correctly, the system does not function properly.

Cisco IOS XR supports RPM signing and signature verification for Cisco IOS XR RPM packages in the ISO and upgrade images. All RPM packages in the Cisco IOS XR ISO and upgrade images are signed to ensure cryptographic integrity and authenticity. This guarantees that the RPM packages have not been tampered with and the RPM packages are from Cisco IOS XR. The private key, used for signing the RPM packages, is created and securely maintained by Cisco.

How to set up a Prompt Interval for the Software Authentication Manager

When the SAM detects an abnormal condition during boot time, it prompts the user to take action and waits for a certain interval. When the user does not respond within this interval, SAM proceeds with a predetermined action that can also be configured.

To set up the Prompt Interval, perform the following tasks.

SUMMARY STEPS

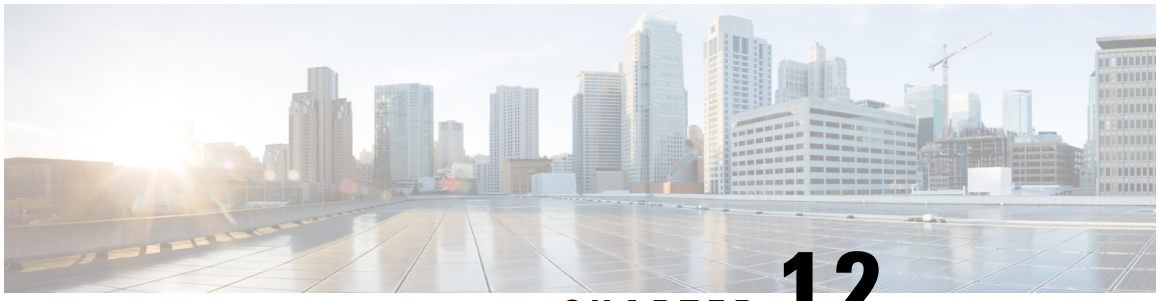
1. **configure**
2. **sam promptinterval** *time-interval* {**proceed** | **terminate**}
3. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**sam promptinterval** *time-interval* {**proceed**	**terminate**} **Example:** `RP/0/RSP0/CPU0:router(config)# sam prompt-interval 25 {proceed | terminate}`	Sets the prompt interval in seconds, after which the SAM either proceeds or terminates the interval. The Prompt interval ranges from 0 to 300 seconds. If the user responds, SAM considers it as a 'Yes' and proceeds with the next action. If the user does not respond, SAM considers it as a 'No' and terminates the action. The default time for which SAM waits is 10 seconds.
Step 3	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session.

	Command or Action	Purpose
		• **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.



CHAPTER 12

Implementing Secure Shell

Secure Shell (SSH) 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 the SSH server are available: SSH Version 1 (SSHv1) and SSH Version 2 (SSHv2). SSHv1 uses Rivest, Shamir, and Adelman (RSA) keys and SSHv2 uses either Digital Signature Algorithm (DSA) keys or Rivest, Shamir, and Adelman (RSA) keys, or Elliptic Curve Digital Signature Algorithm (ECDSA) keys. Cisco IOS XR software supports both SSHv1 and SSHv2.

Note Cisco IOS XR does not support X11 forwarding through an SSH connection.

This module describes how to implement Secure Shell on the the Cisco ASR 9000 Series Router.

Note For a complete description of the Secure Shell commands used in this module , see the *Secure Shell Commands* module in *System Security Command Reference for Cisco ASR 9000 Series Routers*.

Feature History for Implementing Secure Shell

Release	Modification
Release 3.7.2	This feature was introduced.
Release 3.9.0	Support was added for the following enhancements: • RSA based authentication on the SSH server • SFTP client in interactive mode • SFTP server implementation
Release 5.3.0	Support was added for Netconf Subsystem suppport on ssh server using a dedicated port. For more details see chapter *Implementing Network Configuration Protocol* in the *System Management Configuration Guide*.

Release	Modification
Release 6.4.1	Support was added for ECDSA algorithm on IOS-XR SSHv2.
Release 7.0.1	Support was added for SSH configuration option to restrict CIPHER public key and HMAC.
Release 7.0.1	Support was added for automatic host key generation for SSH algorithms.
Release 7.0.1	SSH and SFTP in baseline Cisco IOS XR Software image.
Release 7.0.1	Support was added for enabling CBC mode ciphers 3DES-CBC and AES-CBC for SSHv2 server and client connections.
Release 7.3.1	Support was added for these features: • Ed25519 Public-Key Algorithm Support for SSH • User Configurable Maximum Authentication Attempts for SSH • X.509v3 Certificate-based Authentication for SSH

- Prerequisites for Implementing Secure Shell, on page 348
- SSH and SFTP in Baseline Cisco IOS XR Software Image, on page 349
- Information About Implementing Secure Shell, on page 349
- Guidelines and Restrictions for Implementing Secure Shell, on page 354
- How to Implement Secure Shell, on page 355
- Configuration Examples for Implementing Secure Shell, on page 367
- Multi-channeling in SSH, on page 368
- SSH Configuration Option to Restrict Cipher Public Key and HMAC Algorithm, on page 370
- SSH client strict host key checks, on page 373
- User Configurable Maximum Authentication Attempts for SSH, on page 377
- X.509v3 Certificate-based Authentication for SSH, on page 379
- OpenSSH Certificate based Authentication for Router, on page 387
- Public Key-Based Authentication of SSH Clients, on page 397
- Public key-based Authentication to SSH Server on Routers, on page 401
- Multi-Factor Authentication for SSH, on page 406
- Selective Authentication Methods for SSH Server, on page 411
- SSH Port Forwarding, on page 412
- Non-Default SSH Port, on page 417
- SSH server timeouts, on page 421
- Additional References, on page 428

Prerequisites for Implementing Secure Shell

The following prerequisites are required to implement Secure Shell:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

- Download the required image on your router. The SSH server and SSH client require you to have a a crypto package (data encryption standard [DES], 3DES and AES) from Cisco downloaded on your router.



Note From Cisco IOS XR Software Release 7.0.1 and later, the SSH and SFTP components are available in the baseline Cisco IOS XR software image itself. For details, see, SSH and SFTP in Baseline Cisco IOS XR Software Image, on page 349.

- To run an SSHv2 server, you must have a VRF. This may be the default VRF or a specific VRF. VRF changes are applicable only to the SSH v2 server.
- Configure user authentication for local or remote access. You can configure authentication with or without authentication, authorization, and accounting (AAA). For more information, see the *Authentication, Authorization, and Accounting Commands on* Cisco IOS XR Software module in the *System Security Command Reference for Cisco ASR 9000 Series Routers* publication and *Configuring AAA Services on* Cisco IOS XR Software module in the *System Security Configuration Guide for Cisco ASR 9000 Series Routers* publication.
- AAA authentication and authorization must be configured correctly for Secure Shell File Transfer Protocol (SFTP) to work.

SSH and SFTP in Baseline Cisco IOS XR Software Image

From Cisco IOS XR Software Release 7.0.1 and later, the management plane and control plane components that were part of the Cisco IOS XR security package (k9sec package) are moved to the base Cisco IOS XR software image. These include SSH, SCP, SFTP and IPSec control plane. However, *802.1X protocol (Port-Based Network Access Control) and* data plane components like MACsec and IPsec remain as a part of the security package as per the export compliance regulations. This segreg ation of package components makes the software more modular. It also gives you the flexibility of including or excluding the security package as per your requirements. The new segregation of package components is applicable for both 32 bit and 64 bit IOS XR images.

The base package and the security package allow FIPS, so that the control plane can negotiate FIPS-approved algorithms.

See SSH and SFTP in Baseline Cisco IOS XR Software Image.

Information About Implementing Secure Shell

To implement SSH, you should understand the following concepts:

SSH Server

The SSH server feature enables an SSH client to make a secure, encrypted connection to a Cisco router. 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 IOS XR software

authentication. The SSH server in Cisco IOS XR software works with publicly and commercially available SSH clients.

SSH Client

The SSH client feature is an application running over the SSH protocol to provide device authentication and encryption. The SSH client enables a Cisco router to make a secure, encrypted connection to another Cisco router or to any other device running the SSH server. This connection provides functionality that is similar to that of an outbound Telnet connection except that the connection is encrypted. With authentication and encryption, the SSH client allows for a secure communication over an insecure network.

The SSH client in the Cisco IOS XR software worked with publicly and commercially available SSH servers. The SSH client supported the ciphers of AES, 3DES, message digest algorithm 5 (MD5), SHA1, and password authentication. User authentication was performed in the Telnet session to the router. The user authentication mechanisms supported for SSH were RADIUS, TACACS+, and the use of locally stored usernames and passwords.

The SSH client supports setting DSCP value in the outgoing packets.

```
ssh client dscp <value from 0 - 63>
```

If not configured, the default DSCP value set in packets is 16 (for both client and server).

The SSH client supports the following options:

- DSCP—DSCP value for SSH client sessions.

```
RP/0/5/CPU0:router#configure
RP/0/5/CPU0:router(config)#ssh ?
  client   Provide SSH client service
  server   Provide SSH server service
  timeout  Set timeout value for SSH
RP/0/5/CPU0:router(config)#ssh client ?
```

- Knownhost—Enable the host pubkey check by local database.
- Source-interface—Source interface for SSH client sessions.

```
RP/0/5/CPU0:router(config)#ssh client source-interface ?
  ATM                 ATM Network Interface(s)
  BVI                 Bridge-Group Virtual Interface
  Bundle-Ether        Aggregated Ethernet interface(s)
  Bundle-POS          Aggregated POS interface(s)
  CEM                 Circuit Emulation interface(s)
  GigabitEthernet     GigabitEthernet/IEEE 802.3 interface(s)
  IMA                 ATM Network Interface(s)
  IMtestmain          IM Test Interface
  Loopback            Loopback interface(s)
  MgmtEth             Ethernet/IEEE 802.3 interface(s)
  Multilink           Multilink network interface(s)
  Null                Null interface
  PFItestmain         PFI Test Interface
  PFItestnothw        PFI Test Not-HW Interface
  POS                 Packet over SONET/SDH network interface(s)
  PW-Ether            PWHE Ethernet Interface
  PW-IW               PWHE VC11 IP Interworking Interface
  Serial              Serial network interface(s)
  VASILeft            VASI Left interface(s)
  VASIRight           VASI Right interface(s)
  test-bundle-channel  Aggregated Test Bundle interface(s)
  tunnel-ipsec        IPSec Tunnel interface(s)
  tunnel-mte          MPLS Traffic Engineering P2MP Tunnel interface(s)
```

```
    tunnel-te            MPLS Traffic Engineering Tunnel interface(s)
    tunnel-tp            MPLS Transport Protocol Tunnel interface
  RP/0/5/CPU0:router(config)#ssh client source-interface
  RP/0/5/CPU0:router(config)#
```

- VRF—Source interface VRF for SSH client sessions:

```
  RP/0/5/CPU0:router(config)#ssh client vrf ?
    WORD  VRF name (max:32 chars)
  RP/0/5/CPU0:router(config)#ssh client vrf shan ?
    <cr>
  RP/0/5/CPU0:router(config)#ssh client vrf shan
```

SSH also supports remote command execution as follows:

```
RP/0/5/CPU0:router#ssh ?
  A.B.C.D  IPv4 (A.B.C.D) address
  WORD     Hostname of the remote node
  X:X::X   IPv6 (A:B:C:D...:D) address
  vrf      vrf table for the route lookup
RP/0/5/CPU0:router#ssh 10.1.1.1 ?
  cipher            Accept cipher type
  command           Specify remote command (non-interactive)
  source-interface  Specify source interface
  username          Accept userid for authentication
  <cr>
RP/0/5/CPU0:router#ssh 192.68.46.6 username admin command "show redundancy sum"
Password:


Wed Jan  9 07:05:27.997 PST
    Active Node    Standby Node
    -----------    ------------
       0/4/CPU0        0/5/CPU0 (Node Ready, NSR: Not Configured)

RP/0/5/CPU0:router#
```

SFTP Feature Overview

SSH includes support for standard file transfer protocol (SFTP) , a new standard file transfer protocol introduced in SSHv2. This feature provides a secure and authenticated method for copying router configuration or router image files.

The SFTP client functionality is provided as part of the SSH component and is always enabled on the router. Therefore, a user with the appropriate level can copy files to and from the router. Like the **copy** command, the **sftp** command can be used only in EXEC mode.

The SFTP client is VRF-aware, and you may configure the secure FTP client to use the VRF associated with a particular source interface during connections attempts. The SFTP client also supports interactive mode, where the user can log on to the server to perform specific tasks via the Unix server.

The SFTP Server is a sub-system of the SSH server. In other words, when an SSH server receives an SFTP server request, the SFTP API creates the SFTP server as a child process to the SSH server. A new SFTP server instance is created with each new request.

The SFTP requests for a new SFTP server in the following steps:

- The user runs the **sftp** command with the required arguments
- The SFTP API internally creates a child session that interacts with the SSH server

- The SSH server creates the SFTP server child process
- The SFTP server and client interact with each other in an encrypted format
- The SFTP transfer is subject to LPTS policer "SSH-Known". Low policer values will affect SFTP transfer speeds

Note In IOS-XR SW release 4.3.1 onwards the default policer value for SSH-Known has been reset from 2500pps to 300pps. Slower transfers are expected due to this change. You can adjust the lpts policer value for this punt cause to higher values that will allow faster transfers

When the SSH server establishes a new connection with the SSH client, the server daemon creates a new SSH server child process. The child server process builds a secure communications channel between the SSH client and server via key exchange and user authentication processes. If the SSH server receives a request for the sub-system to be an SFTP server, the SSH server daemon creates the SFTP server child process. For each incoming SFTP server subsystem request, a new SSH server child and a SFTP server instance is created. The SFTP server authenticates the user session and initiates a connection. It sets the environment for the client and the default directory for the user.

Once the initialization occurs, the SFTP server waits for the SSH_FXP_INIT message from the client, which is essential to start the file communication session. This message may then be followed by any message based on the client request. Here, the protocol adopts a 'request-response' model, where the client sends a request to the server; the server processes this request and sends a response.

The SFTP server displays the following responses:

- Status Response
- Handle Response
- Data Response
- Name Response

Note The server must be running in order to accept incoming SFTP connections.

RSA Based Host Authentication

Verifying the authenticity of a server is the first step to a secure SSH connection. This process is called the host authentication, and is conducted to ensure that a client connects to a valid server.

The host authentication is performed using the public key of a server. The server, during the key-exchange phase, provides its public key to the client. The client checks its database for known hosts of this server and the corresponding public-key. If the client fails to find the server's IP address, it displays a warning message to the user, offering an option to either save the public key or discard it. If the server’s IP address is found, but the public-key does not match, the client closes the connection. If the public key is valid, the server is verified and a secure SSH connection is established.

The IOS XR SSH server and client had support for DSA based host authentication. But for compatibility with other products, like IOS, RSA based host authentication support is also added.

RSA Based User Authentication

One of the method for authenticating the user in SSH protocol is RSA public-key based user authentication. The possession of a private key serves as the authentication of the user. This method works by sending a signature created with a private key of the user. Each user has a RSA keypair on the client machine. The private key of the RSA keypair remains on the client machine.

The user generates an RSA public-private key pair on a unix client using a standard key generation mechanism such as ssh-keygen. The max length of the keys supported is 4096 bits, and the minimum length is 512 bits. The following example displays a typical key generation activity:

```
bash-2.05b$ ssh-keygen –b 1024 –t rsa
Generating RSA private key, 1024 bit long modulus
```

The public key must be in base64 encoded (binary) format for it to be imported correctly into the box. You can use third party tools available on the Internet to convert the key to the binary format.

Once the public key is imported to the router, the SSH client can choose to use the public key authentication method by specifying the request using the "-o" option in the SSH client. For example:

```
client$ ssh -o PreferredAuthentications=publickey 1.2.3.4
```

If a public key is not imported to a router using the RSA method, the SSH server initiates the password based authentication. If a public key is imported, the server proposes the use of both the methods. The SSH client then chooses to use either method to establish the connection. The system allows only 10 outgoing SSH client connections.

Currently, only SSH version 2 and SFTP server support the RSA based authentication. For more information on how to import the public key to the router, see the *Implementing Certification Authority Interoperability on the Cisco ASR 9000 Series Router* chapter in this guide.

Note The preferred method of authentication would be as stated in the SSH RFC. The RSA based authentication support is only for local authentication, and not for TACACS/RADIUS servers.

Authentication, Authorization, and Accounting (AAA) is a suite of network security services that provide the primary framework through which access control can be set up on your Cisco router or access server. For more information on AAA, see the *Authentication, Authorization, and Accounting Commands on* the Cisco ASR 9000 Series Router*Software* module in the *System Security Command Reference for Cisco ASR 9000 Series Routers* publication and the *Configuring AAA Services on* the Cisco ASR 9000 Series Router module in the *System Security Configuration Guide for Cisco ASR 9000 Series Routers* publication.

SSHv2 Client Keyboard-Interactive Authentication

An authentication method in which the authentication information is entered using a keyboard is known as keyboard-interactive authentication. This method is an interactive authentication method in the SSH protocol. This type of authentication allows the SSH client to support different methods of authentication without having to be aware of their underlying mechanisms.

Currently, the SSHv2 client supports the keyboard-interactive authentication. This type of authentication works only for interactive applications.



Note The password authentication is the default authentication method. The keyboard-interactive authentication method is selected if the server is configured to support only the keyboard-interactive authentication.

Guidelines and Restrictions for Implementing Secure Shell

The following are some basic SSH guidelines, restrictions, and limitations of the SFTP feature:

- A VRF is not accepted as inband if that VRF is already set as an out-of-band VRF. SSH v1 continues to bind only to the default VRF.
- In order for an outside client to connect to the router, the router needs to have an RSA (for SSHv1 or SSHv2) or DSA (for SSHv2) or ECDSA (for SSHv2) key pair configured. ECDSA, DSA and RSA keys are not required if you are initiating an SSH client connection from the router to an outside routing device. The same is true for SFTP: ECDSA, DSA and RSA keys are not required because SFTP operates only in client mode.
- If you delete all the default crypto keys (the keys with **the_default** label) on the router, the SSH clients cannot establish sessions with the router. Hence, for clients to successfully establish SSH sessions with the router, ensure that at least one default crypto key is always present on the router. In FIPS mode, it is mandatory to have at least one default crypto key of type RSA or ECDSA.
- For SSH sessions, the router supports key-exchange algorithms (**diffie-hellman-group1-sha1** and **curve25519**) and cipher algorithms (**3des-cbc** and **chacha20-poly1305@openssh.com**) only in non-FIPS mode. The SSH session continues to connect if any of these algorithms is pre-configured prior to enabling FIPS mode though.
- In order for SFTP to work properly, the remote SSH server must enable the SFTP server functionality. For example, the SSHv2 server is configured to handle the SFTP subsystem with a line such as **/etc/ssh2/sshd2_config**:
- **subsystem-sftp /usr/local/sbin/sftp-server**
- The SFTP server is usually included as part of SSH packages from public domain and is turned on by default configuration.
- SFTP is compatible with sftp server version OpenSSH_2.9.9p2 or higher.
- RSA-based user authentication is supported in the SSH and SFTP servers. The support however, is not extended to the SSH client.
- Execution shell and SFTP are the only applications supported.
- The AES encryption algorithm is supported on the SSHv2 server and client, but not on the SSHv1 server and client. Any requests for an AES cipher sent by an SSHv2 client to an SSHv1 server are ignored, with the server using 3DES instead.
- The SFTP client does not support remote filenames containing wildcards (*, ?, []). The user must issue the **sftp** command multiple times or list all of the source files from the remote host to download them

on to the router. For uploading, the router SFTP client can support multiple files specified using a wildcard provided that the issues mentioned in the first through third bullets in this section are resolved.

- The cipher preference for the SSH server follows the order AES128, AES192, AES256, and, finally, 3DES. The server rejects any requests by the client for an unsupported cipher, and the SSH session does not proceed.
- Use of a terminal type other than vt100 is unsupported, and the software generates a warning message in this case.
- Password messages of "none" are unsupported on the SSH client.
- Because the router infrastructure does not provide support for UNIX-like file permissions, files created on the local device lose the original permission information. For files created on the remote file system, the file permission adheres to the umask on the destination host and the modification and last access times are the time of the copy.

How to Implement Secure Shell

To configure SSH, perform the tasks described in the following sections:

Configuring SSH

Note For SSHv1 configuration, Step 1 to Step 4 are required. For SSHv2 configuration, Step 1 to Step 4 are optional.

Note From Cisco IOS XR Software Release 7.0.1 and later, the SSH host-key pairs are auto-generated at the time of router boot up. Hence you need not perform steps 5 to 7 to generate the host keys explicitly. See, #unique_359 for details.

SSH server supports setting DSCP value in the outgoing packets.

```
ssh server dscp <value from 0 - 63>
```

If not configured, the default DSCP value set in packets is 16 (for both client and server).

This is the syntax for setting DSCP value:

```
RP/0/5/CPU0:router(config)#ssh server dscp ?
  <0-63>  DSCP value range

RP/0/5/CPU0:router(config)#ssh server dscp 63 ?
  <cr>
RP/0/5/CPU0:router(config)#ssh server dscp 63
RP/0/5/CPU0:router(config)#

RP/0/5/CPU0:router(config)#ssh client dscp ?
  <0-63>  DSCP value range

RP/0/5/CPU0:router(config)#ssh client dscp 0 ?
  <cr>
```

```
RP/0/5/CPU0:router(config)#ssh client dscp 0
RP/0/5/CPU0:router(config)#
```

Perform this task to configure SSH.

SUMMARY STEPS

1. **configure**
2. **hostname** *hostname*
3. **domain name** *domain-name*
4. Use the **commit** or **end** command.
5. **crypto key generate rsa** [**usage keys** | **general-keys**] [*keypair-label*]
6. **crypto key generate dsa**
7. **crypto key generate ecdsa** [**nistp256** | **nistp384** | **nistp521**]
8. **crypto key generate ed25519**
9. **configure**
10. **ssh timeout** *seconds*
11. Do one of the following:
    - **ssh server** [**vrf** *vrf-name* [**ipv4 access-list**IPv4 access-list name] [**ipv6 access-list** IPv6 access-list name]]
    - **ssh server v2**
12. Use the **commit** or **end** command.
13. **show ssh**
14. **show ssh session details**
15. **show ssh history**
16. **show ssh history details**
17. **show tech-support ssh**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**hostname** *hostname* **Example:** `RP/0/RSP0/CPU0:router(config)# hostname router1`	Configures a hostname for your router.
Step 3	**domain name** *domain-name* **Example:**	Defines a default domain name that the software uses to complete unqualified host names.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config)# domain name cisco.com`	
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 5	**crypto key generate rsa** [**usage keys** | **general-keys**] [*keypair-label*] **Example:** `RP/0/RSP0/CPU0:router# crypto key generate rsa general-keys`	Generates an RSA key pair. The RSA key modulus can be in the range of 512 to 4096 bits. • To delete the RSA key pair, use the **crypto key zeroize rsa** command. • This command is used for SSHv1 only.
Step 6	**crypto key generate dsa** **Example:** `RP/0/RSP0/CPU0:router# crypto key generate dsa`	Enables the SSH server for local and remote authentication on the router. The supported key sizes are: 512, 768 and 1024 bits. • The recommended minimum modulus size is 1024 bits. • Generates a DSA key pair. To delete the DSA key pair, use the **crypto key zeroize dsa** command. • This command is used only for SSHv2.
Step 7	**crypto key generate ecdsa** [**nistp256** | **nistp384** | **nistp521**] **Example:** `RP/0/RSP0/CPU0:router# crypto key generate ecdsa nistp256`	Generates an ECDSA key pair. The supported ECDSA curve types are: Nistp256, Nistp384 and Nistp521. • To delete the ECDSA key pair, use the **crypto key zeroize ecdsa [nistp256 | nistp384 | nistp521]** command. • This command is used for SSHv2 only.
Step 8	**crypto key generate ed25519** **Example:** `RP/0/RSP0/CPU0:router# crypto key generate ed25519`	Generates a Ed25519 key pair. To delete the Ed25519 key pair, use the **crypto key zeroize ed25519** command. The support for Ed25519 is available only from Cisco IOS XR Software Release 7.3.1 and later.

	Command or Action	Purpose
Step 9	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 10	**ssh timeout** *seconds* **Example:** `RP/0/RSP0/CPU0:router(config)# ssh timeout 60`	(Optional) Configures the timeout value for user authentication to AAA. • If the user fails to authenticate itself to AAA within the configured time, the connection is terminated. • If no value is configured, the default value of 30 seconds is used. The range is from 5 to 120.

Step 11	Do one of the following:
	• **ssh server** [**vrf** *vrf-name* [**ipv4 access-list**IPv4 access-list name] [**ipv6 access-list** IPv6 access-list name]]
	• **ssh server v2**
	Example:
	`RP/0/RSP0/CPU0:router(config)# ssh`
	or
	`RP/0/RSP0/CPU0:router(config)# ssh server v2`
		• (Optional) Brings up an SSH server using a specified VRF of up to 32 characters. If no VRF is specified, the default VRF is used. To stop the SSH server from receiving any further connections for the specified VRF, use the no form of this command. If no VRF is specified, the default is assumed.Optionally ACLs for IPv4 and IPv6 can be used to restrict access to the server before the port is opened. To stop the SSH server from receiving any further connections for the specified VRF, use the **no** form of this command. If no VRF is specified, the default is assumed.
		Note The SSH server can be configured for mult
		• (Optional) Forces the SSH server to accept only SSHv2 clients if you configure the SSHv2 option by using the **ssh server v2** command. If you choose the **ssh server v2** command, only the SSH v2 client connections are accepted.

Step 12	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

	Command or Action	Purpose
Step 13	**show ssh** **Example:** `RP/0/RSP0/CPU0:router# show ssh`	(Optional) Displays all of the incoming and outgoing SSHv1 and SSHv2 connections to the router.
Step 14	**show ssh session details** **Example:** `RP/0/RSP0/CPU0:router# show ssh session details`	(Optional) Displays a detailed report of the SSHv2 connections to and from the router.
Step 15	**show ssh history** **Example:** `RP/0/RSP0/CPU0:router# show ssh history`	(Optional) Displays the last hundred SSH connections that were terminated.
Step 16	**show ssh history details** **Example:** `RP/0/RSP0/CPU0:router# show ssh history details`	(Optional) Displays the last hundred SSH connections that were terminated with additional details. This command is similar to **show ssh session details** command but also mentions the start and end time of the session.
Step 17	**show tech-support ssh** **Example:** `RP/0/RSP0/CPU0:router# show tech-support ssh`	(Optional) Automatically runs the `show` commands that display system information.

Note The order of priority while doing negotiation for a SSH connection is as follows:

1. ecdsa-nistp-521
2. ecdsa-nistp-384
3. ecdsa-nistp-256
4. rsa
5. dsa

Automatic generation of SSH host-key pairs

An automatic generation of SSH host-key pairs is a security feature that

- creates SSH host-key pairs for supported algorithms (DSA, ECDSA, and RSA) automatically when the router boots,
- eliminates the need for explicit manual key generation after initial setup, and
- ensures SSH clients can connect to the SSH server immediately after bootup with a basic configuration.

Table 44: Feature History Table

Feature Name	Release info	Description
SSH key strength: 3072-bit by default	Release 26.1.1	This update enhances device security by automatically generating RSA 3072-bit SSH host keys during system boot, instead of RSA 2048-bit keys. 3072-bit aligns with industry best practices and provides improved cryptographic protection, ensuring secure SSH access and compliance with the latest security requirements.

The automatic generation feature simplifies device provisioning, especially in zero touch provisioning (ZTP) and Golden ISO boot scenarios. Since SSH host-key pairs are present immediately after boot, administrators do not need to perform manual configurations to enable secure connections.

Starting Cisco IOS XR Software Release 26.1.1, the system automatically generates RSA 3072-bit SSH host keys during boot, replacing the previous default of RSA 2048-bit keys. This enhancement strengthens device security by providing improved cryptographic protection and aligns with the current industry best practices. The default RSA 3072-bit SSH host keys are generated automatically requiring no manual intervention from you.

You can choose specific algorithms to use for SSH host-key pairs by configuring the **ssh server algorithms host-key** command in Global Configuration mode. If certain key pairs are not required, use the **crypto key zeroize** command in EXEC mode to remove them. When upgrading from an earlier software version, the system automatically generates host-key pairs only if they are missing, avoiding duplicate key generation. If SSH host-key pairs are not present after bootup, you can manually generate them using the **crypto key generate** command in EXEC mode.



Note In a system upgrade scenario from version 1 to version 2, the system does not generate the SSH host-key pairs automatically if they were already generated in version 1. The host-key pairs are generated automatically only if they were not generated in version 1.

- On routers with automatic generation enabled, SSH host-key pairs for all supported algorithms are created during initial boot, enabling immediate SSH access by clients.
- During ZTP, there are no additional steps required to configure SSH host-key pairs.

Configure the Allowed SSH Host-Key Pair Algorithms

When the SSH client attempts a connection with the SSH server, it sends a list of SSH host-key pair algorithms (in the order of preference) internally in the connection request. The SSH server, in turn, picks the first matching algorithm from this request list. The server establishes a connection only if that host-key pair is already generated in the system, and if it is configured (using the **ssh server algorithms host-key** command) as the allowed algorithm.

Note If this configuration of allowed host-key pairs is not present in the SSH server, then you can consider that the SSH server allows all host-key pairs. In that case, the SSH client can connect with any one of the host-key pairs. Not having this configuration also ensures backward compatibility in system upgrade scenarios.

Configuration Example

You may perform this (optional) task to specify the allowed SSH host-key pair algorithm (in this example, **ecdsa**) from the list of auto-generated host-key pairs on the SSH server:

```
/* Example to select the ecdsa algorithm */
Router(config)#ssh server algorithms host-key ecdsa-nistp521
```

Similarly, you may configure other algorithms.

Running Configuration

```
ssh server algorithms host-key ecdsa-nistp521
!
```

Verify the SSH Host-Key Pair Algorithms

Note With the introduction of the automatic generation of SSH host-key pairs, the output of the **show crypto key mypubkey** command displays key information of all the keys that are auto-generated. Before its introduction, the output of this show command displayed key information of only those keys that you explicitly generated using the **crypto key generate** command.

```
Router#show crypto key mypubkey ecdsa
Mon Nov 19 12:22:51.762 UTC
Key label: the_default
Type     : ECDSA General Curve Nistp256
Degree   : 256
Created  : 10:59:08 UTC Mon Nov 19 2018
Data     :
04AC7533 3ABE7874 43F024C1 9C24CC66 490E83BE 76CEF4E2 51BBEF11 170CDB26
14289D03 6625FC4F 3E7F8F45 0DA730C3 31E960FE CF511A05 2B0AA63E 9C022482
6E

Key label: the_default
Type     : ECDSA General Curve Nistp384
Degree   : 384
Created  : 10:59:08 UTC Mon Nov 19 2018
Data     :
04B70BAF C096E2CA D848EE72 6562F3CC 9F12FA40 BE09BFE6 AF0CA179 F29F6407
FEE24A43 84C5A5DE D7912208 CB67EE41 58CB9640 05E9421F 2DCDC41C EED31288
6CACC8DD 861DC887 98E535C4 893CB19F 5ED3F6BC 2C90C39B 10EAED57 87E96F78
B6

Key label: the_default
Type     : ECDSA General Curve Nistp521
Degree   : 521
Created  : 10:59:09 UTC Mon Nov 19 2018
```

```
Data     :
0400BA39 E3B35E13 810D8AE5 260B8047 84E8087B 5137319A C2865629 8455928F
D3D9CE39 00E097FF 6CA369C3 EE63BA57 A4C49C02 B408F682 C2153B7F AAE53EF8
A2926001 EF113896 5F1DA056 2D62F292 B860FDFB 0314CE72 F87AA2C9 D5DD29F4
DA85AE4D 1CA453AC 412E911A 419E9B43 0A13DAD3 7B7E88E4 7D96794B 369D6247
E3DA7B8A 5E
```

The following example shows the output for **ed25519**:

```
Router#show crypto key mypubkey ed25519
Wed Dec 16 16:12:21.464 IST
Key label: the_default
Type     : ED25519
Size     : 256
Created  : 15:08:28 IST Tue Oct 13 2020
Data     :
 649CC355 40F85479 AE9BE26F B5B59153 78D171B6 F40AA53D B2E48382 BA30E5A9

Router#
```

Related Topics

#unique_359

Associated Commands

- ssh server algorithms host-key
- show crypto key mypubkey

Ed25519 Public-Key Signature Algorithm Support for SSH

Table 45: Feature History Table

Feature Name	Release Information	Feature Description
Ed25519 Public-Key Algorithm Support for SSH	Release 7.3.1	This algorithm is now supported on Cisco IOS XR 64-bit platforms when establishing SSH sessions. It is a modern and secure public-key signature algorithm that provides several benefits, particularly resistance against several side-channel attacks. Prior to this release, DSA, ECDSA, and RSA public-key algorithms were supported. This command is modified for this feature: ssh server algorithms host-key

This feature introduces the support for Ed25519 public-key algorithm, when establishing SSH sessions, on Cisco IOS XR 64-bit platforms. This algorithm offers better security with faster performance when compared to DSA or ECDSA signature algorithms.

The order of priority of public-key algorithms during SSH negotiation between the client and the server is:

- ecdsa-sha2-nistp256
- ecdsa-sha2-nistp384
- ecdsa-sha2-nistp521
- ssh-ed25519
- ssh-rsa
- ssh-dsa

Restrictions for ED25519 Public Key for SSH

The Ed25519 public key algorithm is not FIPS-certified. That is, if FIPS mode is enabled on the router, the list of public-key algorithms sent during the SSH key negotiation phase does not contain the Ed25519 key. This behavior is applicable only for new SSH connections. Any existing SSH session that has already negotiated Ed25519 public-key algorithm remains intact and continues to execute until the session is disconnected.

Further, if you have configured the router to negotiate only the Ed25519 public-key algorithm (using the **ssh server algorithms host-key** command), and if FIPS mode is also enabled, then the SSH connection to the router fails.

How to Generate Ed25519 Public Key for SSH

To generate Ed25519 public key for SSH, see Generate Crypto Key for Ed25519 Signature Algorithm, on page 175.

You must also specify Ed25519 as the permitted SSH host-key pair algorithm from the list of auto-generated host-key pairs on the SSH server. For details, see Configure the Allowed SSH Host-Key Pair Algorithms, on page 360.

To remove the Ed25519 key from the router, use the **crypto key zeroize ed25519** command in EXEC mode.

Configuring the SSH Client

Perform this task to configure an SSH client.

SUMMARY STEPS

1. **configure**
2. **ssh client knownhost** *device* **:** */filename*
3. Use the **commit** or **end** command.
4. **ssh** {*ipv4-address* | *hostname*} [**username** *user- id* | **cipher des** | **source-interface** *type instance*]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**ssh client knownhost** *device* **:** */filename* **Example:** `RP/0/RSP0/CPU0:router(config)# ssh client knownhost slot1:/server_pubkey`	(Optional) Enables the feature to authenticate and check the server public key (pubkey) at the client end. **Note** The complete path of the filename is require
Step 3	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 4	**ssh** {*ipv4-address* | *hostname*} [**username** *user- id* | **cipher des** | **source-interface** *type instance*] **Example:** `RP/0/RSP0/CPU0:router# ssh remotehost username user1234`	Enables an outbound SSH connection. • To run an SSHv2 server, you must have a VRF. This may be the default or a specific VRF. VRF changes are applicable only to the SSH v2 server. • The SSH client tries to make an SSHv2 connection to the remote peer. If the remote peer supports only the SSHv1 server, the peer internally spawns an SSHv1 connection to the remote server. • The **cipher des** option can be used only with an SSHv1 client. • The SSHv1 client supports only the 3DES encryption algorithm option, which is still available by default for those SSH clients only. • If the *hostname* argument is used and the host has both IPv4 and IPv6 addresses, the IPv6 address is used.

- If you are using SSHv1 and your SSH connection is being rejected, the reason could be that the RSA key pair might have been zeroed out. Another reason could be that the SSH server to which the user is connecting to using SSHv1 client does not accept SSHv1 connections. Make sure that you have specified a hostname and domain. Then use the **crypto key generate rsa** command to generate an RSA host-key pair, and then enable the SSH server.
- If you are using SSHv2 and your SSH connection is being rejected, the reason could be that the DSA, RSA or ECDSA host-key pair might have been zeroed out. Make sure you follow similar steps as mentioned above to generate the required host-key pairs, and then enable the SSH server.
- When configuring the ECDSA, RSA or DSA key pair, you might encounter the following error messages:
  - No hostname specified

  You must configure a hostname for the router using the **hostname** command.
  - No domain specified

  You must configure a host domain for the router using the **domain-name** command.
- The number of allowable SSH connections is limited to the maximum number of virtual terminal lines configured for the router. Each SSH connection uses a vty resource.
- From Cisco IOS XR Release 6.3.1 onwards, the **ssh client enable cipher** command is added for backward compatibility with the older Cisco IOS XR versions.

  For FIPS compliance, in Cisco IOS XR Releases later than 6.2.1, support for weaker ciphers like 3DES and AES CBC was removed and only AES-CTR cipher is supported.
- SSH uses either local security or the security protocol that is configured through AAA on your router for user authentication. When configuring AAA, you must ensure that the console is not running under AAA by applying a keyword in the global configuration mode to disable AAA on the console.



Note If you are using Putty version 0.63 or higher to connect to the SSH client, set the 'Chokes on PuTTYs SSH2 winadj request' option under SSH > Bugs in your Putty configuration to 'On.' This helps avoid a possible breakdown of the session whenever some long output is sent from IOS XR to the Putty client.

Order of SSH Client Authentication Methods

The default order of authentication methods for SSH clients on Cisco IOS XR routers is as follows:

- On routers running Cisco IOS XR SSH:
  - **public-key**, **password** and **keyboard-interactive** (prior to Cisco IOS XR Software Release 24.1.1)
  - **public-key**, **keyboard-interactive** and **password** (from Cisco IOS XR Software Release 24.1.1 and later)
- On routers running CiscoSSH (open source-based SSH):
  - **public-key**, **keyboard-interactive** and **password**

How to Set the Order of Authentication Methods for SSH Clients

To set the preferred order of authentication methods for SSH clients on Cisco IOS XR routers, use the **ssh client auth-method** command in the Global Configuration mode. This command is available from Cisco IOS XR Software Release 7.9.2/Release 7.10.1and later.

Configuration Example

In this example, we set the order of SSH client authentication methods in such a way that public key authentication is negotiated first, followed by keyboard-interactive, and then password-based authentication.

```
Router#configure
Router(config)#ssh client auth-method public-key keyboard-interactive password
Router(config-ssh)#commit
```

Running Configuration

```
Router#show run ssh client auth-methods
Tue Nov 21 17:55:44.688 IST
ssh client auth-methods public-key keyboard-interactive password
Router#
```

Configuring CBC Mode Ciphers

In release 7.0(1), you can enable CBC mode ciphers 3DES-CBC and AES-CBC for SSHv2 server and client connections. The ciphers are disabled by default.

Procedure

Step 1 **configure**

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Enters global configuration mode.

Step 2 **ssh server enable cipher aes-cbc 3des-cbc**

Example:

```
Router(config)# ssh server enable cipher aes-cbc 3des-cbc
```

Step 3 **ssh client enable cipher aes-cbc 3des-cbc**

Example:

```
Router(config)# ssh client enable cipher aes-cbc 3des-cbc
```

Step 4 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.

- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 5 **show ssh session details**

Example:

```
Router# show ssh session details
```

Configuring CBC Mode Ciphers

```
/*Enable CBC mode ciphers 3DES-CBC and AES-CBC */
Router# configure
Router(config)# ssh server enable cipher aes-cbc 3des-cbc
Router(config)# ssh client enable cipher aes-cbc 3des-cbc
Router(config)# commit
```

Verify CBC Mode Cipher Configuration.

```
Router# show ssh session details

Thu Sep  6 10:16:26.346 UTC
SSH version : Cisco-2.0

id  key-exchange        pubkey    incipher    outcipher   inmac         outmac
------------------------------------------------------------------------------
Incoming Session
2   ecdh-sha2-nistp256  ssh-rsa   aes128-cbc  aes128-cbc  hmac-sha2-256 hmac-sha2-256
```

Configuration Examples for Implementing Secure Shell

This section provides the following configuration example:

Configuring Secure Shell: Example

This example shows how to configure SSHv2 by creating a hostname, defining a domain name, enabling the SSH server for local and remote authentication on the router by generating a DSA key pair, bringing up the SSH server, and saving the configuration commands to the running configuration file.

From Cisco IOS XR Software Release 7.0.1 and later, the crypto keys are auto-generated at the time of router boot up. Hence, you need to explicitly generate the host-key pair only if it is not present in the router under some scenarios.

After SSH has been configured, the SFTP feature is available on the router.

```
configure
hostname router1
domain name cisco.com
exit
crypto key generate dsa
configure
```

```
ssh server
end
```

Multi-channeling in SSH

The multi-channeling (also called multiplexing) feature on the Cisco IOS XR software server allows you to establish multiple channels over the same TCP connection. Thus, rather than opening a new TCP socket for each SSH connection, all the SSH connections are multiplexed into one TCP connection. For example, with multiplexing support on your XR software server, on a single SSH connection you can simultaneously open a pseudo terminal, remotely execute a command and transfer a file using any file transfer protocol. Multiplexing offers the following benefits:

- You are required to authenticate only once at the time of creating the session. After that, all the SSH clients associated with a particular session use the same TCP socket to communicate to the server.
- Saves time consumed otherwise wasted in creating a new connection each time.

Multiplexing is enabled by default in the Cisco IOS XR software server. If your client supports multiplexing, you must explicitly set up multiplexing on the client for it to be able to send multi-channel requests to the server. You can use OpenSSH, Putty, Perl, WinSCP, Putty, FileZilla, TTSSH, Cygwin or any other SSH-based tool to set up multiplexing on the client. Configure Client for Multiplexing, on page 369 provides an example of how you can configure the client for multiplexing using OpenSSH.

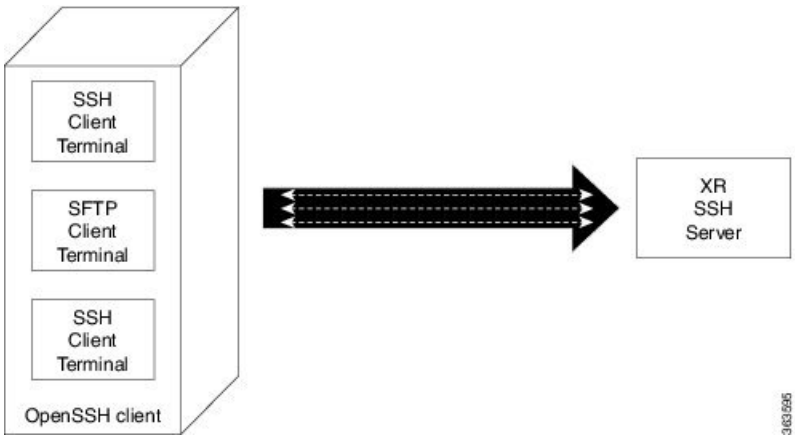
For more information on Multichannel feature, see the Cisco ASR 9000 Series Aggregation Services Router System Security Configuration Guide, Release 5.1.1.

Restrictions for Multi-channeling Over SSH

- Do not use client multiplexing for heavy transfer of data as the data transfer speed is limited by the TCP speed limit. Hence, for a heavy data transfer it is advised that you run multiple SSH sessions, as the TCP speed limit is per connection.
- Client multiplexing must not be used for more than 15 concurrent channels per session simultaneously.
- You must ensure that the first channel created at the time of establishing the session is always kept alive in order for other channels to remain open.
- The **line template default session-limit** command is not supported for SSH.

Client and Server Interaction Over Multichannel Connection

The figure below provides an illustration of a client-server interaction over a SSH multichannel connection.



As depicted in the illustration,

- The client multiplexes the collection of channels into a single connection. This allows different operations to be performed on different channels simultaneously. The dotted lines indicate the different channels that are open for a single session.
- After receiving a request from the client to open up a channel, the server processes the request. Each request to open up a channel represents the processing of a single service.

Note The Cisco IOX software supports server-side multiplexing only.

Configure Client for Multiplexing

The SSH client opens up one TCP socket for all the connections. In order to do so, the client multiplexes all the connections into one TCP connection. Authentication happens only once at the time of creating the session. After that, all the SSH clients associated with the particular session uses the same TCP socket to communicate to the server. Use the following steps to configure client multiplexing using OpenSSH:

SUMMARY STEPS

1. Edit the ssh_config file.
2. Add entries **ControlMaster auto** and **ControlPath**
3. Create a temporary folder.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	Edit the ssh_config file.	Open the ssh_config file with your favorite text editor to configure values for session multiplexing. The system-wide SSH configuration file is located under /etc/ssh/ssh_config. The user configuration file is located under ~/.ssh/config or $HOME/.ssh/config.

	Command or Action	Purpose

Step 2	Add entries **ControlMaster auto** and **ControlPath**
	Example:
	`Host *`
	`ControlMaster auto`
	`ControlPath ~/.ssh/tmp/%r@%h:%p`
		Add the entry ControlMaster auto and ControlPath to the ssh_config file, save it and exit.
		• ControlMaster determines whether SSH will listen for control connections and what to do about them. Setting the ControlMaster to 'auto' creates a primary session automatically but if there is a primary session already available, subsequent sessions are automatically multiplexed.
		• ControlPath is the location for the control socket used by the multiplexed sessions. Specifying the ControlPath ensures that any time a connection to a particular server uses the same specified primary connection.

Step 3	Create a temporary folder.	Create a temporary directory inside the /.ssh folder for storing the control sockets.

SSH Configuration Option to Restrict Cipher Public Key and HMAC Algorithm

The Cisco IOS XR software provides a new configuration option to control the key algorithms to be negotiated with the peer while establishing an SSH connection with the router. With this feature, you can enable the insecure SSH algorithms on the SSH server, which are otherwise disabled by default. A new configuration option is also available to restrict the SSH client from choosing the HMAC, or hash-based message authentication codes algorithm while connecting to the SSH server on the router. You can also configure a list of ciphers as the default cipher list, thereby having the flexibility to enable or disable any particular cipher.

Commands introduced:

- ssh algorithms cipher
- ssh disable hmac



Caution Use caution in enabling the insecure SSH algorithms to avoid any possible security attack.

To disable the HMAC algorithm, use the **ssh client disable hmac** command or **ssh server disable hmac** command in Global Configuration mode.

To enable the required cipher, use the **ssh server enable cipher** command in Global Configuration mode.

The supported encryption algorithms (in the order of preference) are:

1. aes128-ctr
2. aes192-ctr
3. aes256-ctr

4. aes128-gcm@openssh.com
5. aes256-gcm@openssh.com
6. aes128-cbc
7. aes192-cbc
8. aes256-cbc
9. 3des-cbc

In SSH, the CBC-based ciphers are disabled by default. To enable these, you can use the **ssh client enable cipher** command or **ssh server enable cipher** command with the respective CBC options (aes-cbc or 3des-cbc). All CTR-based and GCM-based ciphers are enabled by default.

Disable HMAC Algorithm

Configuration Example to Disable HMAC Algorithm

```
Router(config)# ssh server disable hmac hmac-sha1
Router(config)#commit

Router(config)# ssh client disable hmac hmac-sha1
Router(config)#commit
```

Running Configuration

```
ssh server disable hmac hmac-sha1
!

ssh client disable hmac hmac-sha1
!
```

Related Topics

SSH Configuration Option to Restrict Cipher Public Key and HMAC Algorithm, on page 370

Associated Commands

- **ssh client disable hmac**
- **ssh server disable hmac**

Enable Cipher Public Key

Configuration Example to Enable Cipher Public Key

To enable all ciphers on the client and the server:

Router 1:

```
Router(config)# ssh client algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc
aes128-ctr aes128-cbc aes128-gcm@openssh.com aes256-gcm@openssh.com 3des-cbc
```

Router 2:

```
Router(config)# ssh server algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc
aes128-ctr aes128-cbc aes128-gcm@openssh.com aes256-gcm@openssh.com 3des-cbc
```

To enable the CTR cipher on the client and the CBC cipher on the server:

Router 1:

```
Router(config)# ssh client algorithms cipher aes128-ctr aes192-ctr aes256-ctr
```

Router 2:

```
Router(config)# ssh server algorithms cipher aes128-cbc aes256-cbc aes192-cbc 3des-cbc
```

Without any cipher on the client and the server:

Router 1:

```
Router(config)# no ssh client  algorithms  cipher
```

Router 2:

```
Router(config)# no ssh server  algorithms  cipher
```

Enable only the deprecated algorithms on the client and the server:

Router 1:

```
Router(config)# ssh client algorithms cipher aes128-cbc aes192-cbc aes256-cbc 3des-cbc
```

Router 2:

```
Router(config)# ssh server algorithms cipher aes128-cbc aes192-cbc aes256-cbc 3des-cbc
```

Enable the deprecated algorithm (using **enable cipher** command) and enable the CTR cipher (using **algorithms cipher** command) on the client and the server:

Router 1:

```
Router(config)# ssh client enable cipher aes-cbc 3des-cbc
Router(config)# ssh client algorithms cipher aes128-ctr aes192-ctr aes256-ctr
```

Router 2:

```
Router(config)# ssh server enable cipher aes-cbc 3des-cbc
```

```
Router(config)# ssh server algorithms cipher aes128-ctr aes192-ctr aes256-ctr
```

Running Configuration

All ciphers enabled on the client and the server:

Router 1:

```
ssh client algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc aes128-ctr aes128-cbc
 aes128-gcm@openssh.com aes256-gcm@openssh.com 3des-cbc
!
```

Router 2:

```
ssh client algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc aes128-ctr aes128-cbc
 aes128-gcm@openssh.com aes256-gcm@openssh.com 3des-cbc
!
```

Related Topics

SSH Configuration Option to Restrict Cipher Public Key and HMAC Algorithm, on page 370

Associated Commands

- **ssh client enable cipher**
- **ssh server enable cipher**
- **ssh client algorithms cipher**
- **ssh server algorithms cipher**

SSH client strict host key checks

A SSH client strict host key checking is a feature that

- enforces validation of server host keys using a system-wide `known_hosts` file,
- restricts SSH client connections to hosts with trusted or previously accepted host keys, and
- supports configurable policies for new or changed keys, such as off, accept-new, ask, or yes.

The knownhosts file is located at `/mnt/rdsfs/ciscossh/known_hosts` and is shared across all users, providing persistent, centralized host key management for SSH client operations.

Table 46: Feature History Table

Feature Name	Release Information	Feature Description
SSH client strict host key checking	Release 26.3.1	You enhance SSH security by enforcing strict host key checking, allowing you to control how the SSH client handles new or changed server keys - accept, reject, or prompt for approval. Trusted host keys are stored system-wide and persist across reloads, ensuring consistent validation for your outbound SSH connections.

Restrictions for SSH client strict host key check

Provides reference information about the modes of SSH client strict host key checking, their behaviors, and their effect on host key storage, automation, and system security.

- All SSH client users share the default known hosts file at `/mnt/rdsfs/ciscossh/known_hosts`. Adding, deleting, or accepting a host key in this file affects all SSH client sessions.
- RDSFS must be available to retain host keys across device reloads and switchovers.
- When you use the `ssh-client knownhosts import <file-path>` command, the imported file replaces the contents of the default known-hosts file. The system does not append the imported entries.
- Host keys learned through SSH, SCP, SFTP, or other CLI operations that initiate SSH, SCP, or SFTP sessions are stored only in the default system-wide knownhosts file.
- The following strict host key checking modes are available:
  - `strict-check off`: Automatically accepts new or changed host keys. This mode weakens protection against man-in-the-middle attacks and should be used only for temporary, low-risk troubleshooting.
  - `strict-check accept-new`: Automatically accepts and stores new host keys, but rejects changed host keys.
  - `strict-check ask`: Prompts for approval only when an unknown host key is encountered. After the key is accepted and stored, any later change to that host key is rejected.
  - `strict-check yes`: Rejects both unknown and changed host keys. To avoid connection failures, pre-populate the trusted host list before enabling this policy.



Note If the legacy `ssh client knownhost` command is present in the global configuration, you cannot configure `strict-hostkey-check`. The exec-mode `ssh-client knownhosts` commands are also unavailable until you remove the legacy configuration.

Configure SSH client strict host key checking

Configure SSH client strict host key checking to control how routers verify SSH server host keys for outbound SSH connections and prevent unauthorized access.

- Enforce trust of only verified SSH servers to meet security policies.
- Minimize operational risk by ensuring host trust decisions are policy-driven.

Note The global strict host key checking configuration can be overridden for an individual SSH, SCP, or SFTP session using the `strict-hostkey-check` option.

SSH client strict host key checking validates remote server keys before starting an SSH session. All SSH client operations reference a single, system-wide `knownhosts` file at `/mnt/rdsfs/ciscossh/known_hosts`, regardless of the user.

This mechanism ensures that clients only connect to trusted hosts and enables administrators to choose the level of enforcement (**off**, **accept-new**, **ask**, or **yes**) for their environment.

- No per-user host key management; changes are global for all SSH client users.
- The `strict-check` mode controls the degree of host verification and is set according to deployment security requirements.

Before you begin

- Administrative access to the Cisco router.
- Persistent storage is available and RDSFS is mounted at `/mnt/rdsfs`.
- Your organization's SSH security policies for trusted host validation are known.

Follow these steps to configure SSH client strict host key checking.

Procedure

Step 1 Set the SSH client strict host key checking mode.

Example:

```
Router# configure
Router(config)# ssh client strict-hostkey-check {off | accept-new | ask | yes}
Router(config)# commit
```

- `off`: Automatically accepts new and changed host keys.
- `accept-new`: Accepts and stores new host keys, but rejects changed host keys.
- `ask`: Prompts for an unknown host key; after it is accepted and stored, rejects any later key change.
- `yes`: Rejects both unknown and changed host keys.

Step 2 Exit configuration mode.

Example:

```
Router(config)# end
```

Step 3 For an individual node, use exec-mode commands to manage trusted-host entries.

The following commands operate on the current node. To align trusted-host entries across nodes, copy and import the same baseline known-hosts file on each node.

a) Import a baseline knownhosts file.

Example:

```
Router# ssh-client knownhosts import <file-path>
```

b) Export the system-wide known-hosts database.

Example:

```
Router# ssh-client knownhosts export <file-path>
```

c) Add, delete, or clear trusted-host entries if required.

Example:

```
Router# ssh-client knownhosts add
Router# ssh-client knownhosts delete hostname
Router# ssh-client knownhosts delete line
Router# ssh-client knownhosts clear
```

To manage the trusted host entries, see SSH client knownhosts commands, on page 376.

Step 4 Verify the trusted-host list.

Example:

```
Router# show ssh client knownhosts
```

SSH client strict host key checking is configured. The router handles new and changed SSH server keys according to the selected enforcement mode.

What to do next

Review the router's outbound SSH connection logs to confirm that SSH client connections are handled according to the selected strict host-key checking mode and your organization's SSH security policies.

SSH client knownhosts commands

The trusted-host database is stored in `/mnt/rdsfs/ciscossh/known_hosts` and is shared by all SSH client users.

Use these commands to manage entries in the trusted-host database.

Table 47: SSH client known hosts commands

Command	Description
ssh-client knownhosts add	Adds a trusted-host entry.
ssh-client knownhosts import	Imports a knownhosts file.

Command	Description
ssh-client knownhosts export	Exports the trusted-host database.
ssh-client knownhosts delete hostname	Deletes an entry by hostname.
ssh-client knownhosts delete line	Deletes a numbered entry.
ssh-client knownhosts clear	Removes all entries.
show ssh client knownhosts	Displays the trusted-host entries.

Run these commands in exec mode.

Host keys learned during SSH, SCP, SFTP, or another CLI session are stored only in the system-wide database.

User Configurable Maximum Authentication Attempts for SSH

Table 48: Feature History Table

Feature Name	Release Information	Feature Description
User Configurable Maximum Authentication Attempts for SSH	Release 7.3.1	This feature allows you to set a limit on the number of user authentication attempts allowed for SSH connection, using the three authentication methods that are supported by Cisco IOS XR. The limit that you set is an overall limit that covers all the authentication methods together. If the user fails to enter the correct login credentials within the configured number of attempts, the connection is denied and the session is terminated. This command is introduced for this feature: ssh server max-auth-limit

The three SSH authentication methods that are supported by Cisco IOS XR are public-key (which includes certificate-based authentication), keyboard-interactive, and password authentication. The limit count that you set as part of this feature comes into effect whichever combination of authentication methods you use. The limit ranges from 3 to 20; default being 20 (prior to Cisco IOS XR Software Release 7.3.2, the limit range was from 4 to 20).

Restrictions for Configuring Maximum Authentication Attempts for SSH

These restrictions apply to configuring maximum authentication attempts for SSH:

- This feature is available only for Cisco IOS XR routers functioning as SSH server; not for the ones functioning as SSH clients.
- This configuration is not specific to individual user; the limit remains same for all the users.
- Due to security reasons, the SSH server limits the number of authentication attempts that explicitly uses the password authentication method to a maximum of 3. You cannot change this particular limit of 3 by configuring the maximum authentication attempts limit for SSH.

  For example, even if you configure the maximum authentication attempts limit as 5, the number of authentication attempts allowed using the password authentication method still remain as 3.

Configure Maximum Authentication Attempts for SSH

You can use the **ssh server max-auth-limit** command to specify the maximum number of authentication attempts allowed for SSH connection.

Configuration Example

```
Router#configure
Router(config)#ssh server max-auth-limit 5
Router(config)#commit
```

Running Configuration

```
Router#show running-configuration ssh
ssh server max-auth-limit 5
ssh server v2
!
```

Verification

The system displays the following SYSLOG on the router console when maximum authentication attempts is reached:

```
RP/0/RP0/CPU0:Oct 6 10:03:58.029 UTC: SSHD_[68125]: %SECURITY-SSHD-3-ERR_GENERAL : Max
authentication tries reached-exiting
```

Associated Commands

- **ssh server max-auth-limit**

X.509v3 Certificate-based Authentication for SSH

Table 49: Feature History Table

Feature Name	Release Information	Feature Description
X.509v3 Certificate-based Authentication for SSH	Release 7.3.1	This feature adds new public-key algorithms that use X.509v3 digital certificates for SSH authentication. These certificates use a chain of signatures by a trusted certification authority to bind a public key to the digital identity of the user who is authenticating with the SSH server. These certificates are difficult to falsify and therefore used for identity management and access control across many applications and networks. Commands introduced for this feature are: ssh server certificate ssh server trustpoint This command is modified for this feature: ssh server algorithms host-key

This feature adds new public-key algorithms that use X.509v3 digital certificates for SSH authentication. This feature support is available for the SSH server for server and user authentication.

The X.509v3 certificate-based authentication for SSH feature supports the following public-key algorithms:

- **x509v3-ssh-dss**
- **x509v3-ssh-rsa**
- **x509v3-ecdsa-sha2-nistp256**
- **x509v3-ecdsa-sha2-nistp384**
- **x509v3-ecdsa-sha2-nistp521**



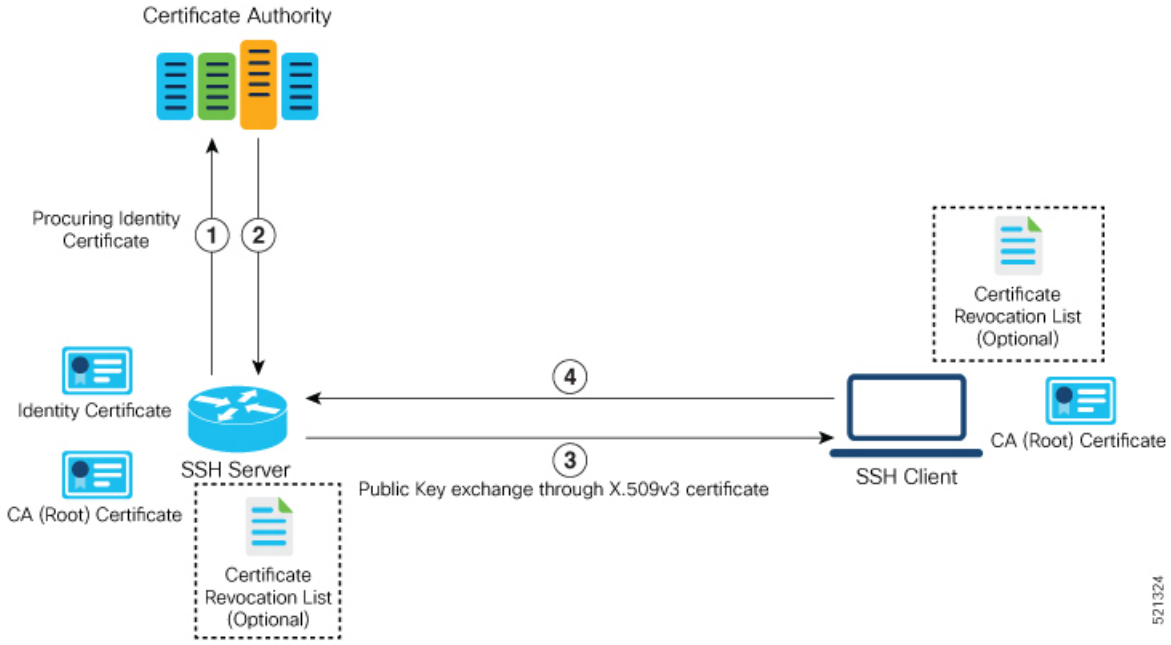
Note While user authentication by using X.509v3 certificate-based authentication for the SSH server is supported using all algorithms listed above, server authentication is supported only with the **x509v3-ssh-rsa** algorithm.

There are two SSH protocols that use public-key cryptography for authentication:

- Transport Layer Protocol (TLP) described in RFC4253—this protocol mandates that you use a digital signature algorithm (called the public-key algorithm) to authenticate the server to the client.
- User Authentication Protocol (UAP) described in RFC4252—this protocol allows the use of a digital signature to authenticate the client to the server (public-key authentication).

For TLP, the Cisco IOS XR SSH server provides its server certificate to the client, and the client verifies the certificate. Similarly, for UAP, the client provides an X.509 certificate to the server. The peer checks the validity and revocation status of the certificate. Based on the result, access is allowed or denied.

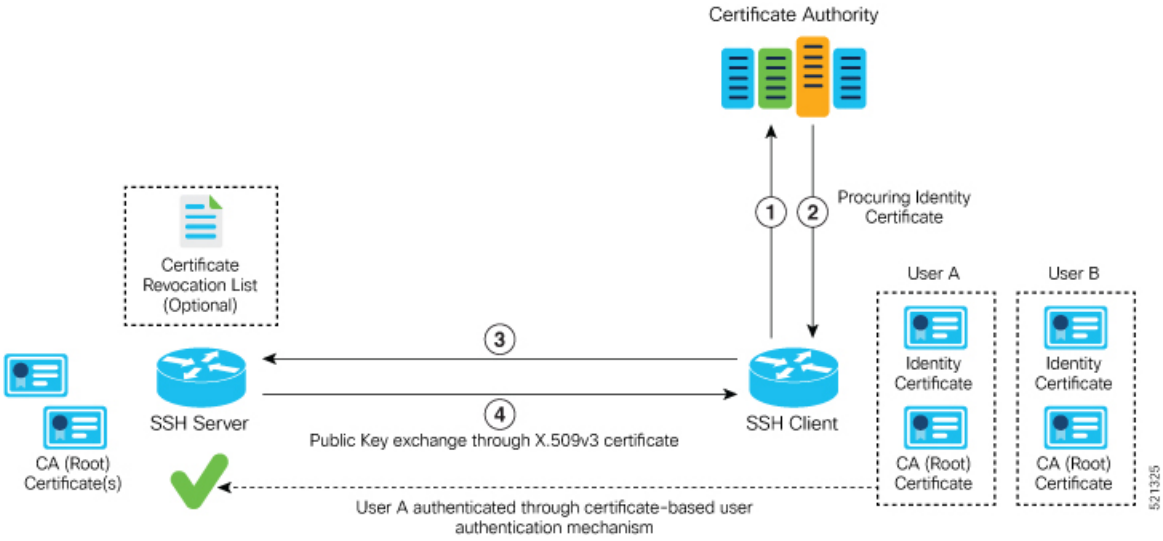
Server Authentication using X.509v3 Certificate



The server authentication process involves these steps:

1. The SSH server procures a valid identity certificate from a well-known certificate authority. This certificate can be obtained manually (through cut-and-paste mechanism) or through protocol implementations such as Simple Certificate Enrollment Protocol (SCEP).
2. The certificate authority provides valid identity certificates and associated root certificates. The requesting device stores these certificates locally.
3. The SSH server presents the certificate to the SSH client for verification.
4. The SSH client validates the certificate and starts the next phase of the SSH connection.

User Authentication using X.509v3 Certificate



The user authentication phase starts after the SSH transport layer is established. At the beginning of this phase, the client sends the user authentication request to the SSH server with required parameters. The user authentication process involves these steps:

1. The SSH client requests a valid identity certificate from a well-known certificate authority.
2. The certificate authority provides valid identity certificates and associated root certificates. The requesting device stores these certificates locally.
3. The SSH client presents the certificate to the SSH server for verification.
4. The SSH server validates the certificate and starts the next phase of the SSH connection.

The certificate-based authentication uses public key as the authentication method. The certificate validation process by the SSH server involves these steps:

- The SSH server retrieves the user authentication parameters, verifies the certificate, and also checks for the certificate revocation list (CRL).
- The SSH server extracts the *username* from the certificate attributes, such as *subject name* or *subject alternate name* (SAN) and presents them to the AAA server for authorization.
- The SSH server then takes the extracted *username* and validates it against the incoming *username* string present in the SSH connection parameter list.

Restrictions for X.509v3 Certificate-based Authentication for SSH

These restrictions apply to the X.509v3 certificate-based authentication feature for SSH:

- Supported only for Cisco IOS XR devices acting as the SSH server; not for the Cisco IOS XR devices acting as the SSH client.
- Supported only for local users because TACACS and RADIUS server do not support public-key authentication. As a result, you must include the **local** option for AAA authentication configuration.

Note Although this feature supports only local authentication, you can enforce remote authorization and accounting using the TACACS server.

- Certificate verification using the Online Certificate Status Protocol (OCSP) is currently not supported. The revocation status of certificates is checked using a certificate revocation list (CRL).
- To avoid user authentication failure, the chain length of the user certificate must not exceed the maximum limit of 9.

Configure X.509v3 Certificate-based Authentication for SSH

To enable X.509v3 certificate-based authentication for SSH, these tasks for server and user authentication:

Server Authentication:

- Configure the list of host key algorithms—With this configuration, the SSH server decides the list of host keys to be offered to the client. In the absence of this configuration, the SSH server sends all available algorithms to the user as host key algorithms. The SSH server sends these algorithms based on the availability of the key or the certificate.
- Configure the SSH trust point for server authentication—With this configuration, the SSH server uses the given trust point certificate for server authentication. In the absence of this configuration, the SSH server does not send **x509v3-ssh-rsa** as a method for server verification. This configuration is not VRF-specific; it is applicable to SSH running in all VRFs.

  The above two tasks are for server authentication and the following ones are for user authentication.

User Authentication:

- Configure the trust points for user authentication—With this configuration, the SSH server uses the given trust point for user authentication. This configuration is not user-specific; the configured trust points are used for all users. In the absence of this configuration, the SSH server does not authenticate using certificates. This configuration is not specific to a VRF; it is applicable to SSH running in all VRFs.

  You can configure up to ten user trust points.
- Specify the *username* to be picked up from the certificate—This configuration specifies which field in the certificate is to be considered as the *username*. The **common-name** from the **subject name** or the **user-principle-name(othername)** from the **subject alternate name**, or both can be configured.
- Specify the maximum number of authentication attempts allowed by the SSH server—The value ranges from 4 to 20. The default value is 20. The server closes the connection if the number of user attempts exceed the configured value.
- AAA authentication configuration—The AAA configuration for public key is the same as that for the regular or keyboard-interactive authentication, except that it mandates local method in the authentication method list.

Configuration Example

In this example, the **x509v3-ssh-rsa** is specified as the allowed host key algorithm to be sent to the client. Similarly, you can configure other algorithms, such as **ecdsa-sha2-nistp521, ecdsa-sha2-nistp384, ecdsa-sha2-nistp256, ssh-rsa**, and **ssh-dsa**.

```
/* Configure the lits of host key algorithms */
Router#configure
Router(config)#ssh server algorithms host-key x509v3-ssh-rsa
Router(config)#commit

/* Configure the SSH trustpoint for server authentication */
Router#configure
Router(config)#ssh server certificate trustpoint host tp1
Router(config)#commit

/* Configure the trustpoints to be used for user authentication */
Router#configure
Router(config)#ssh server trustpoint user tp1
Router(config)#ssh server trustpoint user tp2
Router(config)#commit

/* Specifies the username to be picked up from the certificate.
In this example, it specifies the user common name to be picked up from the subject name
field */
Router#configure
Router(config)#ssh server certificate username common-name
Router(config)#commit

/* Specifies the maximum authentication limit for the SSH server */
Router#configure
Router(config)#ssh server max-auth-limit 5
Router(config)#commit

/* AAA configuration for local authentication with certificate and
remote authorization with TACACS+ or RADIUS */
Router#configure
Router(config)#aaa authentication login default group tacacs+ local
Router(config)#aaa authorization exec default group radius group tacacs+
Router(config)#commit
```

Running Configuration

```
ssh server algorithms host-key x509v3-ssh-rsa
!
ssh server certificate trustpoint host tp1
!
ssh server trustpoint user tp1
ssh server trustpoint user tp2
!
ssh server certificate username common-name
!
ssh server max-auth-limit 5
!
```

Verification of Certificate-based Authentication for SSH

You can use the **show ssh server** command to see various parameters of the SSH server. For certificate-based authentication for SSH, the **Certificate Based** field displays *Yes*. Also, the two new fields, **Host Trustpoint** and **User Trustpoints**, display the respective trust point names.

```
Router#show ssh server
Wed Feb 19 15:23:38.752 IST
---------------------
SSH Server Parameters
---------------------

Current supported versions := v2
                  SSH port := 22
                  SSH vrfs := vrfname:=default(v4-acl:=, v6-acl:=)
              Netconf Port := 830
              Netconf Vrfs := vrfname:=default(v4-acl:=, v6-acl:=)

 Algorithms
---------------
        Hostkey Algorithms := x509v3-ssh-rsa,
ecdsa-sha2-nistp521,ecdsa-sha2-nistp384,ecdsa-sha2-nistp256,ssh-rsa,ssh-dsa
   Key-Exchange Algorithms :=
ecdh-sha2-nistp521,ecdh-sha2-nistp384,ecdh-sha2-nistp256,diffie-hellman-group14-sha1
     Encryption Algorithms :=
aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes256-gcm@openssh.com
            Mac Algorithms := hmac-sha2-512,hmac-sha2-256,hmac-sha1

 Authetication Method Supported
------------------------------------
                 PublicKey := Yes
                  Password := Yes
      Keyboard-Interactive := Yes
         Certificate Based := Yes

 Others
------------
                     DSCP  := 16
                Ratelimit  := 60
             Sessionlimit  := 100
                Rekeytime  := 60
       Server rekeyvolume  := 1024
  TCP window scale factor  := 1
            Backup Server  := Enabled, vrf:=default, port:=11000
Host Trustpoint            := tp1
User Trustpoints           := tp1 tp2
```

You can use the **show ssh session details** command to see the chosen algorithm for an SSH session:

```
Router#show ssh session details
Wed Feb 19 15:33:00.405 IST
SSH version : Cisco-2.0

id      key-exchange           pubkey               incipher    outcipher   inmac
outmac
-----------------------------------------------------------------------------------------------

Incoming Sessions
1       ecdh-sha2-nistp256     x509v3-ssh-rsa       aes128-ctr  aes128-ctr  hmac-sha2-256
hmac-sha2-256
```

Similarly, you can use the **show ssh** command to verify the authentication method used. In this example, it shows as *x509-rsa-pubkey*:

```
Router#show ssh
Sun Sep 20 18:14:04.122 UTC
SSH version : Cisco-2.0

id  chan pty  location   state        userid    host          ver authentication connection
 type
-----------------------------------------------------------------------------------------------
Incoming sessions
4   1    vty0 0/RP0/CPU0 SESSION_OPEN  9chainuser 10.105.230.198 v2  x509-rsa-pubkey
Command-Line-Interface

Outgoing sessions
```

SYSLOGS

You can observe relevant SYSLOGS on the router console in various scenarios listed here:

- On successful verification of peer certificate:

```
RP/0/RP0/CPU0:Aug 10 15:01:34.793 UTC: locald_DLRSC[133]: %SECURITY-PKI-6-LOG_INFO :
Peer certificate verified successfully
```

- When user certificate CA is not found in the trust point:

```
RP/0/RP0/CPU0:Aug  9 22:06:43.714 UTC: locald_DLRSC[260]: %SECURITY-PKI-3-ERR_GENERAL
: issuer not found in trustpoints configured
RP/0/RP0/CPU0:Aug  9 22:06:43.714 UTC: locald_DLRSC[260]: %SECURITY-PKI-3-ERR_ERRNO :
Error:='Crypto Engine' detected the 'warning' condition 'Invalid trustpoint or trustpoint
 not exist'(0x4214c000), cert verificationn failed
```

- When there is no CA certificate or host certificate in the trust point:

```
RP/0/RP1/CPU0:Aug 10 00:23:28.053 IST: SSHD_[69552]: %SECURITY-SSHD-4-WARNING_X509 :
could not get the host cert chain, 'sysdb' detected the 'warning' condition 'A SysDB
client tried to access a nonexistent item or list an empty directory', x509 host auth
will not be used
RP/0/RP1/CPU0:Aug 10 00:23:30.442 IST: locald_DLRSC[326]: %SECURITY-PKI-3-ERR_ERRNO :
Error:='Crypto Engine' detected the 'warning' condition 'Invalid trustpoint or trustpoint
 not exist'(0x4214c000), Failed to get trustpoint name from
```

How to Disable X.509v3 Certificate-based Authentication for SSH

- Server Authentication — You can disable X.509v3 certificate-based server authentication for SSH by using the **ssh server algorithms host-key** command. From the list of auto-generated host-key pairs algorithms on the SSH server, this command configures allowed SSH host-key pair algorithms. Hence, if you have this configuration without specifying the **x509-ssh-rsa** option in the preceding command, it is equivalent to disabling the X.509v3 certificate-based server authentication for the SSH server.
- User Authentication — You can remove the user trust point configuration (**ssh server trustpoint user**) so that the SSH server does not allow the X.509v3 certificate-based authentication.

Failure Modes for X.509v3 Certificate-based Authentication for SSH

If the **ssh server certificate trustpoint host** configuration is missing, or if the configuration is present, but the router certificate is not present under the trust point, then the SSH server does not add **x509-ssh-rsa** to the list of supported host key methods during key exchange.

Also, the user authentication fails with an error message if:

- User certificate is in an incorrect format.
- The chain length of the user certificate is more than the maximum limit of 9.
- Certificate verification fails due to any reason.

Related Topics

- X.509v3 Certificate-based Authentication for SSH, on page 379

Associated Commands

- **ssh server algorithms hostkey**
- **ssh server certificate username**
- **ssh server max-auth-limit**
- **ssh server trustpoint host**
- **ssh server trustpoint user**
- **show ssh server**
- **show ssh session details**

OpenSSH Certificate based Authentication for Router

Table 50: Feature History Table

Feature Name	Release Information	Feature Description
OpenSSH Certificate based Authentication for Router	Release 7.5.3	You can now use OpenSSH certificates to authenticate to the remote routers from a client machine. This feature uses the ssh-keygen utility, a standard SSH component to generate and manage authentication keys, available in OpenSSH to create a CA (Certificate Authority) like infrastructure for logging into the router. In this feature, the certificates that are used to authenticate router and client are both signed by the same CA. This automatically establishes trust between router and client, and eliminates the need to establish trust, while using the client for remote logging to router for the first time.

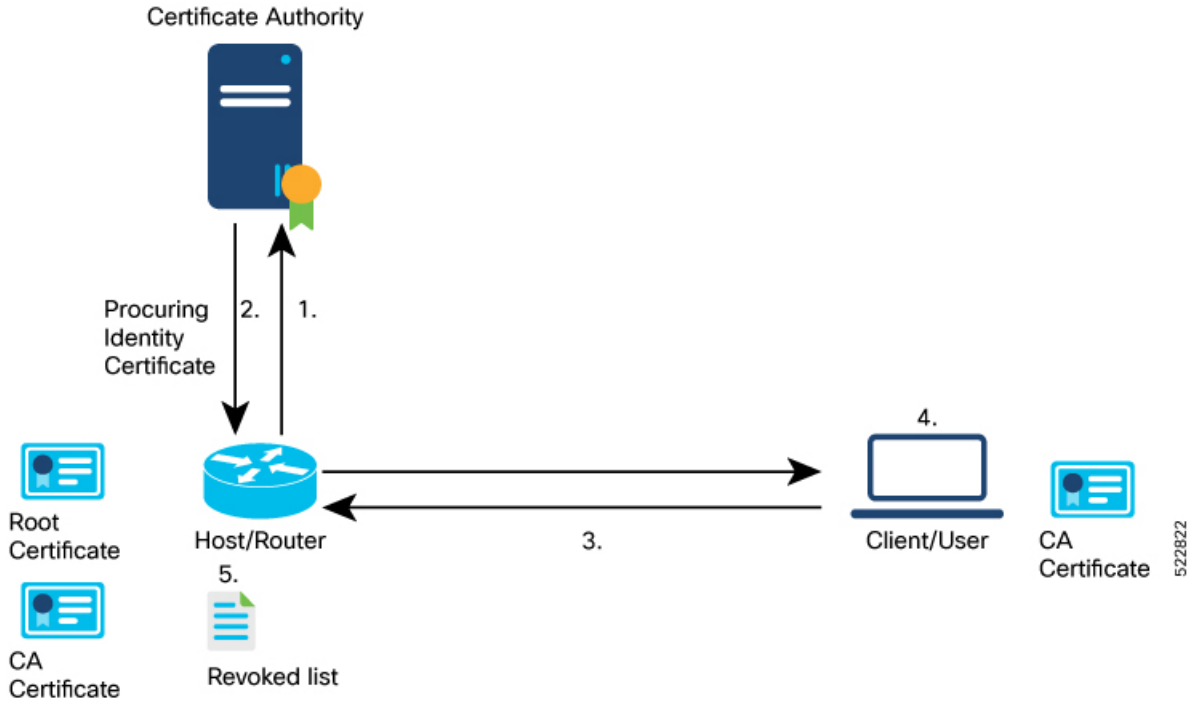
OpenSSH is the open-source implementation of the SSH Protocol. In OpenSSH certificate-based authentication, you can use the ssh-keygen utility to create a certificate signing infrastructure. A digital certificate with public and private key pair, created using the ssh-keygen utility, authenticates the host and the user certificates. The user certificate authenticates the client machine to the router. The client machine is a system that the user utilizes to establish remote access to the router. When a user attempts to log in to the router using the client machine, the client machine presents its certificate to the router. The router checks for the identity and validity of the certificate to decide whether to allow or deny the connection request. The host certificate in the router authenticates the router to the client. Overall, the host and user certificates together establish a two-way secure communication channel.

The OpenSSH based authentication for the router has the following major phases:

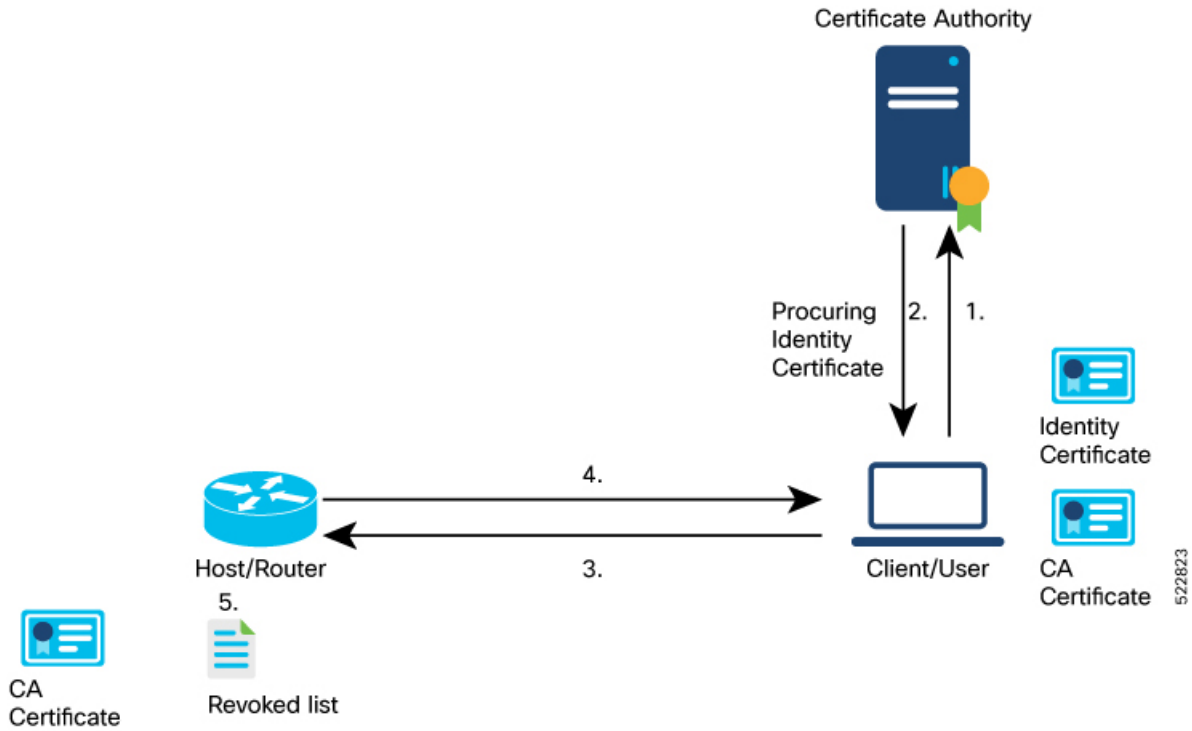
Establishing the trustpoints: In the router, you must create a trustpoint and configure the router to use this trustpoint for the host and user authentication. You can have a same or different trustpoints for these entities. While the router can have only one trustpoint, the user can have up to ten trustpoints.

Creating the CA: Any system with the OpenSSH feature acts like the CA. The ssh-keygen creates the CA certificate and utilizes it to sign the router and user certificate.

Router authentication: You must copy the CA public key in the CA server to the router and ensure to create a CSR (Certificate Signing Request) in the router. The CSR file is further copied to the CA server and signed using the CA certificate. The CA signed certificate is copied back to the router to complete its authentication with CA.



User authentication: You must create a digital certificate for the user using the ssh-keygen utility and sign the public key using the CA certificate. The CA signed user certificate must be copied to the client system using which you would log into the router using the specified user.



Remote access to the router: After the host and user authentication, you can access the router using SSH in the client system that is used to authenticate the user.

Feature Highlights

- OpenSSH certificates use the Certificate Authority (CA) infrastructure to act as a trusted entity while signing the host or user certificates.
- OpenSSH certificates contain a public and private key pair, including identity and validity information. These are signed using a standard SSH public key using the ssh-keygen utility.
- The router certificate includes information such as the host public key, public key of the signing CA, type (host), certificate validity, Key ID, serial number of the certificate, and so on.
- The user certificate contains the user's public key, the public key of the signing CA, Key ID, type (user), serial number, certificate validity, principals matched against the login username, and so forth.
- The CA is just another SSH key created using the ssh-keygen utility. However, rather than utilizing it for authenticating the router or user directly, it's used to sign and validate the other keys that are used for authenticating the router and the user.
- You can view the router and user certificate properties using the ssh-keygen.
- The OpenSSH certificates support the following encryptions:
  - RSA
  - DSA
  - ECDSA
  - ED25519

Prerequisites

- You must have a client machine which has OpenSSH feature with the **ssh-keygen** utility to act as CA.

Configuration Example

The following high-level steps help you set up OpenSSH based Authentication:

1. Create a trustpoint in the router and configure the router to use this trustpoint for the host and user authentication.
2. Creating CA, the CA here is a dedicated system with OpenSSH feature that provides a certificate signing infrastructure using the ssh-keygen utility.
3. Host authentication, the host here is the Cisco IOS XR router.
4. User authentication, a user is any entity attempting to access the router. Generally refers to system to access the router CLI remotely. User is also referred to as client.
5. Access the router in the client using the OpenSSH authentication

This section contains the detailed procedure to enable this feature in your router.

1. Create a trustpoint in the router and configure the router to use this trustpoint for the host and user authentication.

   a. [Router Config mode] Create a trustpoint in the router.

   ```
   Router# config
   Router(config)# crypto ca openssh trustpoint test
   Router(config)# commit
   ```

   b. [Router Config mode] Configure the trustpoint for host authentication.

   ```
   Router# config
   Router(config)# ssh server openssh trustpoint host test
   Router(config)# commit
   ```

   c. [Router Config mode] Configure the trustpoint for user authentication

   ```
   Router# config
   Router(config)# ssh server openssh trustpoint user test
   Router(config)# commit
   ```

2. Creating CA

   a. [CA Server] In the dedicated machine with OpenSSH feature to act as CA, generate a certificate using the **ssh-keygen** utility:

   ```
   [root@CAServer test]# ssh-keygen -t rsa -f cacert
   Generating public/private rsa key pair.
   Enter passphrase (empty for no passphrase):
   Enter same passphrase again:
   Your identification has been saved in cacert.
   Your public key has been saved in cacert.pub.
   The key fingerprint is:
   SHA256:/B2b8V7jKXwGphf75fkO74U/mpuHgDHmvF4okexdKhY root@CAServer
   The key's randomart image is:
   +---[RSA 2048]----+
   |                 |
   |                 |
   |                 |
   |      ...+       |
   |       ES +.o    |
   |      . +=+o X . |
   |       = +o.O O.+|
   |      . o... B+@*|
   |        ..  .=XBX|
   +----[SHA256]-----+

   [root@CAServer test]# ls
   cacert  cacert.pub
   ```

Note Leave the passphrase empty.

3. Host (Router) authentication

   a. [CA Server] Open the CA public key from CA server and copy it contents.

   ```
   [root@CAServer test]# cat cacert.pub
   ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCigl/zhyjuGOBYz5bu+GL76
   HBaROV0pVS4Lx3pf1jcjrFkVibPKKkVeX/lE7sZIJ0anU9vYSJZW8zrl8z06G
   ```

```
qzmnJqRRaXa9vfwNmjvNdRwxuBA3Uk/G1sbmcusMXBXoY6z0IEMhlVN0hCqE4
cIFgLxgHpYAaqyl2hISaomTCNhkbD770Ot8zbyRjl6G0Ps0ggYHWmfLZf/tbF
IBPWpuuuA3LvpZIiTaztevQaWYSyK22h3tp3K62IOBX3gUd4Yr+Gvo4PNA26e
21cUE2aVJsl6J9MeFITR2NzY1cmZ44KWi6bglkPlE4KBiRsbHCvs4wlaUaO5q
hNj1BdH3/Hha4x root@CAServer
```

b. [Router EXEC mode] Add the contents of the CA public key to router trustpoint.

```
Router#crypto ca openssh authenticate test
Enter the CA pubkey.
End with a blank line or the word "quit" on a line by itself
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCigl/zhyjuGOBYz5bu+GL7
6HBaROV0pVS4Lx3pf1jcjrFkVibPKKkVeX/lE7sZIJ0anU9vYSJZW8zrl8z0
6GqzmnJqRRaXa9vfwNmjvNdRwxuBA3Uk/G1sbmcusMXBXoY6z0IEMhlVN0hC
qE4cIFgLxgHpYAaqyl2hISaomTCNhkbD770Ot8zbyRjl6G0Ps0ggYHWmfLZf
/tbFIBPWpuuuA3LvpZIiTaztevQaWYSyK22h3tp3K62IOBX3gUd4Yr+Gvo4P
NA26e21cUE2aVJsl6J9MeFITR2NzY1cmZ44KWi6bglkPlE4KBiRsbHCvs4wl
aUaO5qhNj1BdH3/Hha4x root@CAServer
Do you accept this certificate? [yes/no]: yes
```

c. [Router EXEC mode] Validate the copied CA public key by viewing the OpenSSH certificates in the CA trustpoint configured in the router.

```
Router#show crypto ca openssh certificates
Fri Sep 16 06:59:38.347 UTC

Trustpoint       : test
================================================================
CA certificate
================================================================

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCigl/zhyjuGOBYz5bu+GL76HBa
ROV0pVS4Lx3pf1jcjrFkVibPKKkVeX/lE7sZIJ0anU9vYSJZW8zrl8z06GqzmnJq
RRaXa9vfwNmjvNdRwxuBA3Uk/G1sbmcusMXBXoY6z0IEMhlVN0hCqE4cIFgLxgHp
YAaqyl2hISaomTCNhkbD770Ot8zbyRjl6G0Ps0ggYHWmfLZf/tbFIBPWpuuuA3Lv
pZIiTaztevQaWYSyK22h3tp3K62IOBX3gUd4Yr+Gvvcjdvjwevfo4PNA26e21cUE
2aVJsl6J9eFITR2NzY1cmZ44KWi6bglkPlE4KBiRsbHCvs4wlaUaO5qhNj1BdH3/
Hha4x root@CAServer
```

d. [Router EXEC mode] Generate a CSR for the CA public key in the router.

```
Router#crypto ca openssh enroll test
Fri Sep 16 06:34:41.230 UTC
Display Certificate Request to terminal? [yes/no]: yes
---Hostkey follows---

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCaXqjc45LohfiHJ1iq8sSpaJmdR
QQJo6bRMhkdxY1pbjEYrwjPTn5SnC1NZYwsTPSHlbYBxQRLBHLv80Gbb0v+uJ1T0T
4tAmLgSYPXaHqYIyepCeMKSkSKLgZ0Pf+oGBMtf3uUuLqCgnFAwjrzDBXJYfF+bd/
ieXMwKKNH3YiceLOqe4BAYRU6m+wiuZ8is+bIfy32Eq7gWuPUz8XpxaCt3icpqfrj
7/vm7amKf1GpiheaRJH0Cg4JAmJpAQkuPjx+Y9SZw2yTJP+IKr9tSoSWyiHo2B/Yg
3yERd7M8dQEsvrGy5KIf92x+eLPlGl5gB9ykEPDUpXeaYTu5wtDR/Jd

---End - This line not part of hostkey---
Redisplay enrollment request? [yes/no]: n
```

e. [Router EXEC mode] Select the hostkey contents of the CSR file and copy the hostkey of the CSR.

f. [CA server] Create a .pub file in the CA server for the CSR hostkey and paste the copied hostkey contents in this file.

```
[root@CAServer test]# vim host.pub
/* Here we are using the vim text editor to create the host.pub file */
/* You can use any text editor of your choice  */
```

g. [CA server] Execute the following block to sign the CSR file using the CA certificate

```
[root@CAServer test]# ssh-keygen -h -s cacert -I "server" -V +10w -z 10 host.pub
Signed host key host-cert.pub: id "server" serial 10 valid from 2022-09-16T12:26:00
 to 2022-11-25T12:27:17
```

Note Use the following command to sign the CSR file using the CA certificate:

ssh-keygen -h -s <CACert> -I <IdentityOfCSRSys> -V <CertValidity> -z <CertSerialNo> <CopiedCSRFile>

Parameter	Description
CACert	Specify the filename of the CA Server private key
CertValidity	Specify the validity period for the certificate.
CertSerialNo	Specify a serial number for the certificate.
CopiedCSRFile	Specify the name of the file created to copy the contents of CSR in the router.

h. [CA server] Open the signed host certificate and copy the contents.

```
[root@CAServer test]# cat host-cert.pub
ssh-rsa-cert-v01@openssh.com AAAAHHNzaC1yc2EtY2VydC12MDFAb3BlbnNza
C5jb20AAAAgzv0OXl42NNK9C4PtLZniRwBk5jbeS8quNhzVKsRpO7UAAAADAQABAAA
BAQCaXqjc45LohfiHJ1iq8sSpaJmdRQQJo6bRMhkdxY1pbjEYrwjPTn5SnC1NZYwsT
PSHlbYBxQRLBHLv80Gbb0v+uJ1T0T4tAmLgSYPXaHqYIyepCeMKSkSKLgZ0Pf+oGBM
tf3uUuLqCgnFAwjrzDBXJYfF+bd/ieXMwKKNH3YiceLOqe4BAYRU6m+wiuZ8is+bIf
y32Eq7gWuPUz8XpxaCt3icpqfrj7/vm7amKf1GpiheaRJH0Cg4JAmJpAQkuPjx+Y9S
Zw2yTJP+IKr9tSoSWyiHo2B/Yg3yERd7M8dQEsvrGy5KIf92x+eLPlGl5gB9ykEPDU
pXeaYTu5wtDR/JdAAAAAAAAAAoAAAACAAAABnNlcnZlcgAAAAAAAAAYyQeAAAAAAB
jgGdNAAAAAAAAAAAAAAAAAAAABFwAAAAdzc2gtcnNhAAAAAwEAAQAAAQEAooJf84co7
hjgWM+W7vhi++hwWkTldKVUuC8d6X9Y3I6xZFYmzyipFXl/5RO7GSCdGp1Pb2EiWVv
M65fM9Ohqs5pyakUWl2vb38DZo7zXUcMbgQN1JPxtbG5nLrDFwV6GOs9CBDIZVTdIQ
qhOHCBYC8YB6WAGqspdoSEmqJkwjYZGw++9DrfM28kY5ehtD7NIIGB1pny2X/7WxSA
T1qbrrgNy76WSIk2s7Xr0GlmEsittod7adyutiDgV94FHeGK/hr6ODzQNunttXFBNm
lSbJeifTHhSE0djc2NXJmeOCloum4JZD5ROCgYkbGxwr7OMJWlGjuaoTY9QXR9/x4W
uMQAAAQ8AAAAHc3NoLXJzYQAAAQAIywc9o2OWzFq32MnE9IZVVRRiItdXaMVE1EvYu
G92JK7wnMJd50M6QDyfkNmGF4ramF90/bVQpl3UYJzVxCJSEodAq6OmlG3zx/MVayT
unMwV2Fq75PpaoZVpyEKx4kLKA6rNU5Tmbht2OfMQKFvIWyxTDmeLFMvnpt8R0Yrz4
sG5EP1+4E3WthfzZr42Mq2LQJt6aBeYHZDZSp++j7RpA7+T/6n1aGtAjtDIKprOQuE
1higCZmdI+kUZDOXjMJlPmJAnV8fdtnnEpYCyzYeD+rSSF7dlDVrTaiFdqrfCXh+uY
jR1E621sP7UEJOWeiBqSDTJxSRdRBNZq9TLmgJH host.pub
```

i. [Router EXEC mode] Import the signed host certificate to the router.

```
Router# crypto ca openssh import test certificate
/* This command opens the CA trustpoint and you must paste the contents of signed
certificate copied from the CA server */
Fri Sep 16 07:00:27.573 UTC


Enter the OpenSSH certificate.
End with a blank line

ssh-rsa-cert-v01@openssh.com AAAAHHNzaC1yc2EtY2VydC12MDFAb3BlbnNzaC
5jb20AAAAgzv0OXl42NNK9C4PtLZniRwBk5jbeS8quNhzVKsRpO7UAAAADAQABAAABA
```

```
QCaXqjc45LohfiHJ1iq8sSpaJmdRQQJo6bRMhkdxY1pbjEYrwjPTn5SnC1NZYwsTPSH
lbYBxQRLBHLv80Gbb0v+uJ1T0T4tAmLgSYPXaHqYIyepCeMKSkSKLgZ0Pf+oGBMtf3u
UuLqCgnFAwjrzDBXJYfF+bd/ieXMwKKNH3YiceLOqe4BAYRU6m+wiuZ8is+bIfy32Eq
7gWuPUz8XpxaCt3icpqfrj7/vm7amKf1GpiheaRJH0Cg4JAmJpAQkuPjx+Y9SZw2yTJ
P+IKr9tSoSWyiHo2B/Yg3yERd7M8dQEsvrGy5KIf92x+eLPlGl5gB9ykEPDUpXeaYTu
5wtDR/JdAAAAAAAAAAoAAAACAAAABnNlcnZlcgAAAAAAAAAAYyQeAAAAABjgGdNAAA
AAAAAAAAAAAAAAAABFwAAAAdzc2gtcnNhAAAAAwEAAQAAAQEAooJf84co7hjgWM+W7v
hi++hwWkTldKVUuC8d6X9Y3I6xZFYmzyipFXl/5RO7GSCdGp1Pb2EiWVvM65fM9Ohqs
5pyakUWl2vb38DZo7zXUcMbgQN1JPxtbG5nLrDFwV6GOs9CBDIZVTdIQqhOHCBYC8YB
6WAGqspdoSEmqJkwjYZGw++9DrfM28kY5ehtD7NIIGB1pny2X/7WxSAT1qbrrgNy76W
SIk2s7Xr0GlmEsittod7adyutiDgV94FHeGK/hr6ODzQNunttXFBNmlSbJeifTHhSE0
djc2NXJmeOCloum4JZD5ROCgYkbGxwr7OMJWlGjuaoTY9QXR9/x4WuMQAAAQ8AAAAHc
3NoLXJzYQAAAQAIywc9o2OWzFq32MnE9IZVVRRiItdXaMVE1EvYuG92JK7wnMJd50M6
QDyfkNmGF4ramF90/bVQpl3UYJzVxCJSEodAq6OmlG3zx/MVayTunMwV2Fq75PpaoZV
pyEKx4kLKA6rNU5Tmbht2OfMQKFvIWyxTDmeLFMvnpt8R0Yrz4sG5EP1+4E3WthfzZr
42Mq2LQJt6aBeYHZDZSp++j7RpA7+T/6n1aGtAjtDIKprOQuE1higCZmdI+kUZDOXjM
JlPmJAnV8fdtnnEpYCyzYeD+rSSF7dlDVrTaiFdqrfCXh+uYjR1E621sP7UEJOWeiBq
SDTJxSRdRBNZq9TLmgJH host.pub
```

j. [Router EXEC mode] Verify the host certificate import in the router.

```
Router#show crypto ca openssh certificates
Fri Sep 16 07:00:49.488 UTC

Trustpoint       : test
================================================================
CA certificate
================================================================
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCigl/zhyjuGOBYz5bu+GL76HBaROV
0pVS4Lx3pf1jcjrFkVibPKKkVeX/lE7sZIJ0anU9vYSJZW8zrl8z06GqzmnJqRRaXa9
vfwNmjvNdRwxuBA3Uk/G1sbmcusMXBXoY6z0IEMhlVN0hCqE4cIFgLxgHpYAaqyl2hI
SaomTCNhkbD770Ot8zbyRjl6G0Ps0ggYHWmfLZf/tbFIBPWpuuuA3LvpZIiTaztevQa
WYSyK22h3tp3K62IOBX3gUd4Yr+Gvo4PNA26e21cUE2aVJsl6J9MeFITR2NzY1cmZ44
KWi6bglkPlE4KBiRsbHCvs4wlaUaO5qhNj1BdH3/Hha4x root@CAServer

Router certificate
================================================================
Type         : Host Certificate
Key ID       : server
Serial       : 10
Valid        : from Fri Sep 16 06:56:00 2022 to Fri Nov 25 06:57:17 2022
```

4. User authentication

a. [Client machine] Generate an SSH key pair in the client system using the **ssh-keygen** utility for the user.

```
[root@userclient test]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): /root/openssh_client/test/user
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/openssh_client/test/user.
Your public key has been saved in /root/openssh_client/test/user.pub.
The key fingerprint is:
SHA256:rNmS7P0u6l1pm75Kb4KhMxZThwaJ/AMnA9C//Z1GVEY root@userclient.cisco.com
The key's randomart image is:
+---[RSA 2048]----+
|++ . .     .E    |
|  B +       o    |
|   B . .   o     |
|    + +.. .      |
|     * .S.       |
|    +.o=  ..     |
```

```
|     +*+oo+.     |
|    =..=++=o     |
|   . ++.+XO.     |
+----[SHA256]-----+
[root@userclient test]# ls
user  user.pub
```

b. [Client machine] Open the SSH public key file.

Note Copy the public key content for the user certificate.

```
[root@userclient test]# cat user.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCspUNwiwlEy0VXQ1Ruh2peRnAP12LSICNe9
H76xyBiCIXFLLXHTUZZM+W/Pa97pg3fObxaqyNYaeojfwmGeNyPLS9Ha0mqRuLmVCT/1got5I
Rn1AZhufZz7iz1AdW8DMC//KUnUS/T+cEwGrZ//sbIPTMsQZhhaQVk9xqFp9ghPMxwar3vaHa
t9NL6ThrR+viue9IOY5LKMeRnqrf2GFX3L6gHfcgYv9fQOKxI11WjTA645rQyB+NumVlrG6KI
as/xmBCEFHpChGZ1/GSB/atrKeVEWqzsJkpQHXEtE7hwK8gMrL+ad38mbV2Zz6Cc7KHJFEWaZ
sfjFscCP0kzU1gX root@userclient.cisco.com
```

c. [CA server] Create a .pub file in the CA server for the user certificate public key and paste the public key contents from the previous step in this file.

```
[root@CAServer test]# vim user.pub
/* Here we are using the vim text editor to create the user.pub file */
/* You can use any text editor of your choice  */
```

d. [CA server] Sign the user public key using the CA certificate private key.

```
[root@CAServer test]# ssh-keygen -s cacert -I "user" -V +10w -n testuser -z 20 user.pub

Signed user key user-cert.pub: id "user" serial 20 valid from 2022-09-16T12:42:00 to
 2022-11-25T12:43:24
```

Note The command to sign the CSR file using the CA certificate:

ssh-keygen -s <CACert> -I <IdentityOfSysReqCert> -V <CertValidity> -n <Username> -z <CertSerialNo> <CopiedUserCertName>

Note In addition to the mandatory fields specified for the user certificate, you can also configure critical options and extensions for the user certificate. For detailed information on the critical options and extensions, refer ssh-keygen.

Parameter	Description
CACert	Specify the filename of the CA Server private key/
IdentityOfSysReqCert	Specify the identity of the certificate as User
CertValidity	Specify the validity period for the certificate.

Parameter	Description
<Username>	Specify the principals that you want to add to the certificate. **Note** During authentication to the router, the principal in the user certificate is matched against the login username and requests with matching principal and username are permitted for further communication. **Note** You can have multiple principals that are associated with the same certificate. The principals must be separated by commas in the **IdentityOfSysReqCert** field in command to sign the user certificate file using CA certificate.
CertSerialNo	Specify a serial number for the certificate.
CopiedUserCertName	Specify the name of the file created to copy the contents of the user certificate file in the client machine.

e. [CA server] Open the signed user certificate in the CA server and copy the contents.

```
[root@CAServer test]# cat user-cert.pub
ssh-rsa-cert-v01@openssh.com AAAAHHNzaC1yc2EtY2VydC12MDFAb3BlbnNzaC5jb20AA
AAg6xlcZNQTKmUO27dHFcUCk7UzVCPWFMCep7Ldb4lBF6MAAAADAQABAAABAQCspUNwiwlEy0V
XQ1Ruh2peRnAP12LSICNe9H76xyBiCIXFLLXHTUZZM+W/Pa97pg3fObxaqyNYaeojfwmGeNyPL
S9Ha0mqRuLmVCT/1got5IRn1AZhufZz7iz1AdW8DMC//KUnUS/T+cEwGrZ//sbIPTMsQZhhaQV
k9xqFp9ghPMxwar3vaHat9NL6ThrR+viue9IOY5LKMeRnqrf2GFX3L6gHfcgYv9fQOKxI11WjT
A645rQyB+NumVlrG6KIas/xmBCEFHpChGZ1/GSB/atrKeVEWqzsJkpQHXEtE7hwK8gMrL+ad38
mbV2Zz6Cc7KHJFEWaZsfjFscCP0kzU1gXAAAAAAAAABQAAAABAAAABHVzZXIAAAAAAAAAAGMkI
cAAAAAAY4BrFAAAAAAAAACCAAAAFXBlcm1pdC1YMTEtZm9yd2FyZGluZwAAAAAAAAAXcGVybWl
0LWFnZW50LWZvcndhcmRpbmcAAAAAAAAAFnBlcm1pdC1wb3J0LWZvcndhcmRpbmcAAAAAAAAAC
nBlcm1pdC1wdHkAAAAAAAAADnBlcm1pdC11c2VyLXJjAAAAAAAAAAAAAAAEXAAAAB3NzaC1yc2E
AAAADAQABAAABAQCigl/zhyjuGOBYz5bu+GL76HBaROV0pVS4Lx3pf1jcjrFkVibPKKkVeX/lE
7sZIJ0anU9vYSJZW8zrl8z06GqzmnJqRRaXa9vfwNmjvNdRwxuBA3Uk/G1sbmcusMXBXoY6z0I
EMhlVN0hCqE4cIFgLxgHpYAaqyl2hISaomTCNhkbD770Ot8zbyRjl6G0Ps0ggYHWmfLZf/tbFI
BPWpuuuA3LvpZIiTaztevQaWYSyK22h3tp3K62IOBX3gUd4Yr+Gvo4PNA26e21cUE2aVJsl6J9
MeFITR2NzY1cmZ44KWi6bglkPlE4KBiRsbHCvs4wlaUaO5qhNj1BdH3/Hha4xAAABDwAAAAdzc
2gtcnNhAAABABKOHeuTo9OMg6K+HjASpRXD7rQgiiOdljKdkpw4FZlwCOdBegQwPQkFYTNHmrH
frQYY72ZINCAjseq+ZSUCkCqJjyXbvY+ZdmRyy76pQvjitgolZjppJqX38nz3uqz/81A/ZuJiF
811sgJF0Loj7XDN9wjF/zBtsxsXPp7R5c775dmmFgZWQHbSWDlNmnPd9vLZMyBwId//+HV/bCF
LjbqI/nr/amLVjcI0liOZXzsH7bcLFBSDZ3Epd6IAqFEe+URqvscjaaghcvnshvcafdgfaruO0
wedsZX53/pEBKhlGacsachFa+S2QuYqTafqnEtkvJoNKVe7UDq/R4kEXM1s9CclIMOficYJm5L
as+ALR4= root@CAServer.cisco.com
```

f. [CA server] Create a .pub file in the client machine fo the CA signed user certificate and past the signed certificate contents in this file.

```
[root@CAServer test]# vim user-cert.pub
/* Here we are using the vim text editor to create the user-cert.pub file */
/* You can use any text editor of your choice  */
```

g. [Client machine] View the user certificate in the client machine.

```
[root@userclient test]# ssh-keygen -Lf user-cert.pub
user-cert.pub:
        Type: ssh-rsa-cert-v01@openssh.com user certificate
        Public key: RSA-CERT SHA256:rNmS7P0u6l1pm75Kb4KhMxZThwaJ/AMnA9C//Z1GVEY
        Signing CA: RSA SHA256:/B2b8V7jKXwGphf75fkO74U/mpuHgDHmvF4okexdKhY
        Key ID: "user"
        Serial: 20
        Valid: from 2022-09-16T12:44:00 to 2022-11-25T12:45:51
        Principals:
                testuser
        Critical Options: (none)
        Extensions:
                permit-X11-forwarding
                permit-agent-forwarding
                permit-port-forwarding
                permit-pty
                permit-user-rc
```

h. [Client machine] Open the known hosts file in the client system and add the public key of the CA to this file.

Note Add the CA public key to the known hosts file in the following format:

@cert-authority <hostname> <CA Public Key>

```
cat testuser@192.0.2.2 /root/.ssh/known_hosts
@cert-authority ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCigl/zhyjuGOBYz5bu
+GL76HBaROV0pVS4Lx3pf1jcjrFkVibPKKkVeX/lE7sZIJ0anU9vYSJZW8zrl8z06GqzmnJq
RRaXa9vfwNmjvNdRwxuBA3Uk/G1sbmcusMXBXoY6z0IEMhlVN0hCqE4cIFgLxgHpYAaqyl2h
ISaomTCNhkbD770Ot8zbyRjl6G0Ps0ggYHWmfLZf/tbFIBPWpuuuA3LvpZIiTaztevQaWYSy
K22h3tp3K62IOBX3gUd4Yr+Gvo4PNA26e21cUE2aVJsl6J9MeFITR2NzY1cmZ44KWi6bglkP
lE4KBiRsbHCvs4wlaUaO5qhNj1BdH3/Hha4x root@CAServer.cisco.com
```

i. [Router Config mode] Configure the username in the router

```
Router# config
Router(config)# username testuser
Router(config-un)# group root-lr
Router(config-un)# commit
```

5. [Client machine] Access the router in the client using the OpenSSH certificate.

```
[root@userclient test]# ssh -o CertificateFile=user-cert.pub -i user testuser@192.0.2.2
 -o StrictHostKeyChecking=yes
Router#
```

Note The command to access the router in the client machine remotely:

ssh -o CertificateFile=<CA_Signed_User_Certificate_Name> -i <User_Certificate_Private_Key> <Username >@<Router_IP> -o StrictHostKeyChecking=yes

Public Key-Based Authentication of SSH Clients

Table 51: Feature History Table

Feature Name	Release Information	Feature Description

Public Key-Based Authentication of SSH Clients on Cisco IOS XR Routers	Release 7.10.1	You are now assured of cryptographic strength even as you avail of automated password-less login while establishing SSH connections with the server. With the password and keyboard-interactive authentication, Cisco IOS XR routers configured as SSH clients now support public key-based authentication. In this authentication method, passwords need not be sent over the network; hence, it provides an additional layer of security and aids in automation processes. This feature is available only for users locally configured on the router; not those configured on remote servers.
		
		Previous releases supported SSH public key-based authentication only for Cisco IOS XR routers configured as SSH servers.
		
		The feature introduces these changes:
		
		• **CLI**:
		  • **crypto key generate authentication-ssh rsa**
		  • **crypto key zeroize authentication-ssh rsa**
		  • **show crypto key mypubkey authentication-ssh rsa**
		
		• **Yang Data Models**:
		  New Xpaths for:
		  • `Cisco-IOS-XR-crypto-act.yang`
		  • `Cisco-IOS-XR-crypto-oepki-new-oper.yang`
		
		  (see GitHub, YANG Data Models Navigator)

Cisco IOS XR routers configured as SSH clients supported only password authentication and keyboard-interactive authentication for establishing SSH connection with the SSH server. Whereas those IOS XR routers that are configured as SSH servers supported public key-based user authentication as well. From

Cisco IOS XR Software Release 7.10.1 and later, you can use public-key based user authentication for Cisco IOS XR routers configured as SSH clients as well. This feature thereby allows you to use password-less authentication for secure file transfer and copy operations using SFTP and SCP protocols.

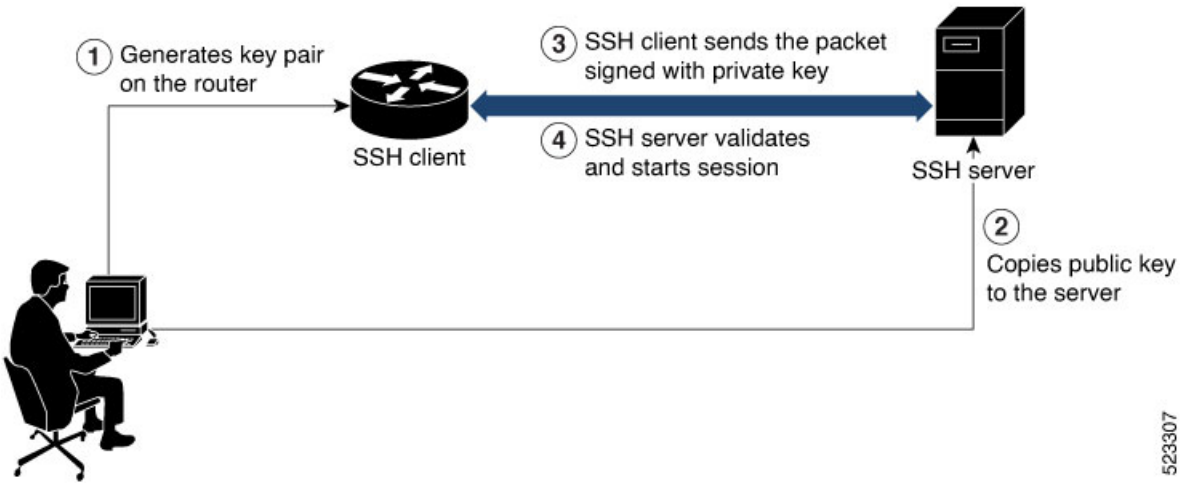
Remote AAA servers such as RADIUS and TACACS+ servers do not support public-key based authentication. Hence this functionality is available only for users who are configured locally on the router and not for users who are configured remotely.

How Does it Work

Public key encryption algorithm works with two keys—a public key and a private key. These keys form a key pair that is specific to a user. They are cryptographically related. The public key is used to encrypt the data and the private key is used to decrypt the data. The data encrypted by the SSH server that holds the public key can then only be read by the entity who holds the corresponding private key.

This image shows the work flow of public key-based authentication of SSH clients.

Figure 17: Public Key-Based Authentication of SSH Clients: Work Flow



You can generate the key pair on the router that is configured as the SSH client. Once it is generated, copy the public key to the SSH server that the user wants to connect to. When the user tries to log in to the server, the SSH client sends a connection request to the SSH server. The SSH server allows access only to users who can confirm that they have the corresponding private key. For this. the SSH server uses the public key of the user to issue a challenge that can be rightly answered by the SSH client using the corresponding private key. The SSH client thus automatically authenticates the user who is logging in to the server using the unique copy of the private key. This process thereby establishes a secure SSH connection to the server in a way that does not require the user to enter the password each time.

Enable Public Key-Based Authentication of SSH Client

Guidelines

These guidelines apply to enabling public key-based SSH authentication on Cisco IOS XR routers that are configured as SSH clients.

- Supports only RSA key.

- Remote AAA servers such as RADIUS and TACACS+ servers do not support public key-based authentication. Hence this functionality is available only for users who are configured locally on the router and not for users who are configured remotely.
- A user with root privileges has permission to create and delete keys for other users.
- If authentication keys are not created, then the SSH client does not proceed with public key-based authentication.
- If user adds the incorrect public key in the SSH server, then the user authentication fails.

Configuration Example

Establishing SSH connection using public key-based authentication on SSH client involves these high-level tasks:

1. Generate RSA key pair on the router that is configured as the SSH client.

   Use the **cyrpto key generate authentication-ssh rsa** command to generate the RSA key pair:

```
Router#crypto key generate authentication-ssh rsa
Wed Dec 21 10:02:57.684 UTC
The name for the keys will be: cisco
  Choose the size of the key modulus in the range of 512 to 4096. Choosing a key modulus
 greater than 512 may take a few minutes.

How many bits in the modulus [2048]:
Generating RSA keys ...
Done w/ crypto generate keypair
[OK]

Router#
```

2. View the details of the generated key.

   Use the **show crypto key mypubkey authentication-ssh rsa** command to view the details of the RSA key. The key value starts with *ssh-rsa* in this output.

```
Router#show crypto key mypubkey authentication-ssh rsa
Wed Dec 21 10:24:34.226 UTC
Key label: cisco
Type     : RSA Authentication
Size     : 2048
Created  : 10:02:59 UTC Wed Dec 21 2022
Data     :
 30820122 300D0609 2A864886 F70D0101 01050003 82010F00 3082010A 02820101
 00A292B0 E45ACBB9 47B9EDA8 47E4664E 58FC3EA5 CE0F6B7A 3C6B7A73 537E6CEB
 .
 .
 .
 FF6BAF95 D9617CF6 65C058CC 7C6C22A9 9E48CC43 FDFF0EB7 ABADEB77 55A274DB
 15020301 0001

OpenSSH Format:
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCikrDkWsu5R7ntqEfkZk5Y/.../2uvldlhfPZlwFjMfGwiqZ5IzEP9/w63q63rd1WidNsV

Router#
```

3. Copy the RSA public key from the SSH client to the SSH server.

You can do this either by logging in to the remote SSH server with your established user credentials, or have a system administrator on the remote system add the key on the SSH server.

If the SSH server is a Cisco IOS XR router, then you can use the **crypto key import authentication rsa** command on the router prompt of the server to import the key from the SSH client. You will then be prompted to enter the public key.

If the SSH server is a Linux server, then you must add the public key to the `~/.ssh/authorized_keys` file of the respective user account in that server. This file contains a list of all authorized public keys on that server.

4. The user configured on the SSH client can now log in to the remote SSH server (*209.165.200.225* in this example) without providing the user account password.

```
Router#ssh user1@209.165.200.225
```

This process establishes a successful SSH connection between the client and the server using public key-based authentication.

How to Delete the SSH Public Keys

Use the **crypto key zeroize authentication-ssh rsa username** command to delete the RSA keys.

```
Router#crypto key zeroize authentication-ssh rsa username user1
```

Public key-based Authentication to SSH Server on Routers

Table 52: Feature History Table

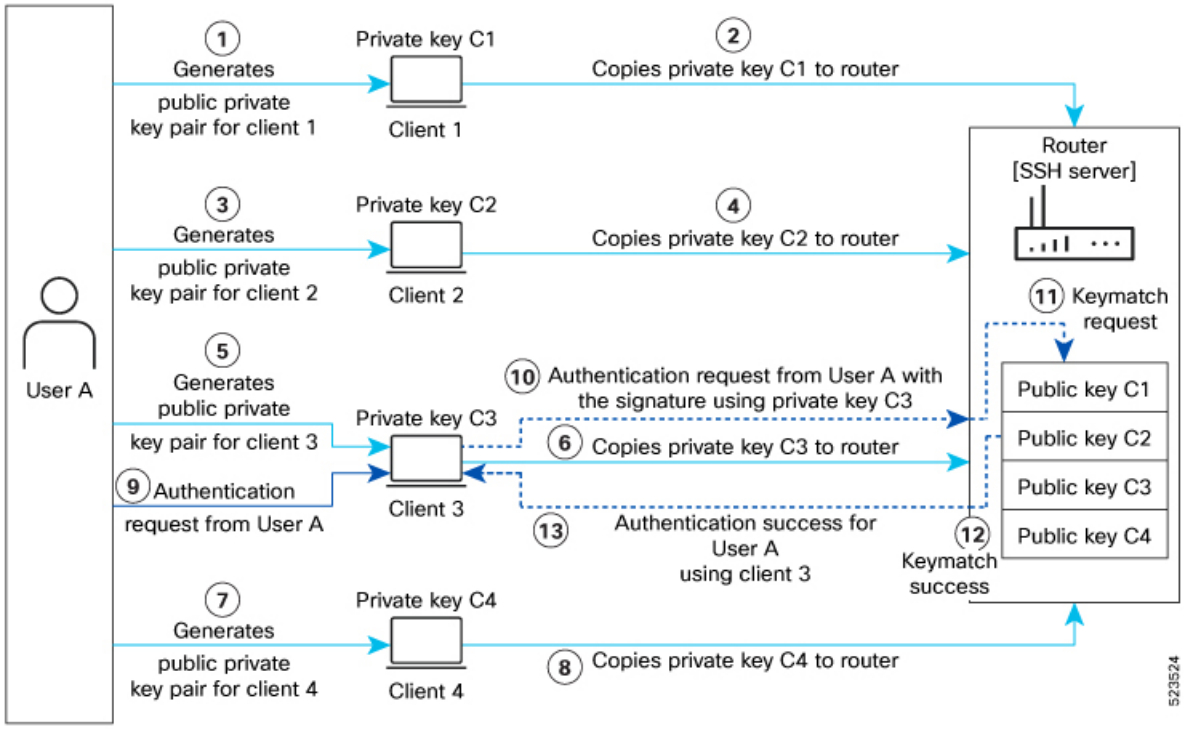
Feature Name	Release Information	Feature Description
Multiple Public Keys per User for Public Key-based Authentication	Release 7.11.1	We provide greater flexibility to access secure routers by allowing four public keys to be used for authentication. With the ability to associate multiple public keys with your user account on the router, we've also simplified the authentication process by eliminating the need to create unique users for each SSH client device. The feature introduces these changes: **CLI:** • The **second**, **third**, and **fourth** keywords are introduced in the **crypto key import authentication rsa** command. • The **second**, **third**, and **fourth** keywords are introduced in the **crypto key zeroize authentication rsa** command. • The **second**, **third**, and **fourth** keywords are introduced in the **keystring** command.`  **YANG Data Models:**  • `Cisco-IOS-XR-crypto-act`  • `Cisco-IOS-XR-um-ssh-cfg` (See GitHub, YANG Data Models Navigator)

Public key-based authentication provides password-less authentication to the routers. In this method, the user authentication relies on a cryptographic key pair: a public key and a private key. The user generates a key pair in the client device using utilities such as ssh-keygen. The public key is imported and stored in the router(SSH server), while the private key is in the user device(SSH client). While attempting public key-based authentication from the client, the user presents a signature created using the private key to the router. The router verifies the authenticity of that signature using the public key associated with that user in its database.

The authentication is successful when the signature matches the public key and user access is permitted. Otherwise, the authentication fails, and the router denies the user access. With public key-based authentication, the routers offer a more secure authentication method than traditional password-based authentication because it is less vulnerable to brute force attacks and password theft.

From Cisco IOS XR Software Release 7.11.1, the routers support up to four public keys per user for public key-based authentication to the routers. Previously, the users could have a single key pair. This constraint restricts an individual user in the router from having multiple systems to access the routers. Also, it necessitates creating a unique user in the router for each device to be an authorized SSH client to the router. However, the multiple public keys per user for public key-based authentication feature overcome these restrictions by allowing up to four public keys per user. As a result, the users can employ any corresponding private key to access the router.

Figure 18: Multiple Public Keys per User for Public Key-based Authentication



You can generate the key pair on each of the SSH clients. You must then copy the public keys to the router. When the user tries to log in to the router, the SSH client sends a connection request with a signature created using the private key. The router then checks the authenticity of the request by matching it against the public keys associated with that user in its database. Suppose one of the multiple public keys associated with that user matches the signature; the router authenticates such user, confirming that the user has the corresponding private key. This process thereby establishes a secure SSH connection to the router.

Guidelines and Restrictions for Public key-based authentication to Routers

- You can add public keys by importing the public key file or directly adding the public keystring to the router.
- The maximum number of public keys supported per user is four.

- The router supports importing or adding only one public key at a time. Even though the router supports up to four keys per user, you can only import or add them to the router one after the other and not simultaneously.
- To import the public key files to the router, use the **crypto key import authentication rsa** command.
- The router supports importing public keys in the following formats:
  - RSA
  - Base 64
  - PEM PKCS1
  - PEM PKCS8
- To delete the public key files in the router, use the **crypto key zeroize authentication rsa** command.
- You can import the public keys using the **crypto key import authentication rsa** command in the Global Configuration mode and EXEC mode. However, use the same operation mode to import and delete the public keys. That is, if you import the public keys in the Global Configuration mode, delete such keys in Global Configuration mode only. Similarly, if you import the public keys in the EXEC mode, delete such keys in EXEC mode only.
- You can use SSH configurations to add or delete a public key in the router.
- The router supports only the RSA key format while using SSH configurations to add a public key for public key-based authentication to the router.

Configure Public key-based Authentication to Routers

This section details different methods of enabling flexible public key-based authentication and importing public keys to the router:

Configurations

Using public-key import:

1. [Router] Create a user in the router:

```
Router# config
Router(config)# username testuser1
Router(config)# commit
```

2. [Client] Generate RSA key pairs on the SSH clients.
3. [Router] Copy the public keys from the clients to the router.



Note You can skip step 3 while using the tftp filepath in step 4. For more details, **crypto key import authentication rsa** command.

4. [Router] Import public keys to the router:

```
Router# configure
Router(config)# crypto key import authentication rsa username testuser1
```

```
disk0:/id_rsa_key1.pub
Router(config)# crypto key import authentication rsa username testuser1 second
disk0:/id_rsa_key2.pub
Router(config)# crypto key import authentication rsa username testuser1 third
disk0:/id_rsa_key3.pub
Router(config)# crypto key import authentication rsa username testuser1 fourth
disk0:/id_rsa_key4.pub
Router(config)# commit
```

You can now access the router from any of the four SSH clients using the same user.

5. [Client] Access the router in the client:

```
[root@userclient test]# ssh testuser1@192.0.2.2
```

Using SSH configurations:

1. [Router] Create a user in the router:

```
Router# config
Router(config)# ssh server username testuser2
Router((config-user-key))# commit
```

2. [Client] Generate RSA key pair on the SSH clients.

3. [Router] Add public keys from the SSH clients for a user to the router:

```
Router# configure
Router(config)# ssh server username testuser2
Router(config-user-key)# keystring ssh-rsa
[redacted]
Router(config-user-key)# keystring ssh-rsa second
[redacted]
Router(config-user-key)# keystring ssh-rsa third
[redacted]
Router(config-user-key)# keystring ssh-rsa fourth
[redacted]
Router(config)# commit
```

You can now access the router from any of the four SSH clients using the same user.

4. [Client] Access the router in the client:

```
[root@userclient test]# ssh testuser2@192.0.2.2
```

Verification

Public-key import:

```
Router# show crypto key authentication rsa testuser1 all
Wed Sep 20 16:28:09.114 IST
Key label: testuser1firstkey
Type     : RSA Signature
Size     : 768
Created  : 16:27:54 IST Wed Sep 20 2023
Data     :
 307C300D 06092A86 4886F70D 01010105 00036B00 30680261 00BDD9A2 B8D61FA3
 AED1B6EC FB975512 32BFE99E 65FDCC01 FA14956C 7B06C2A5 CEE9E637 56FE38F6
 878ED2F4 CD1C1F28 3F535F23 9F5F8763 19BA0269 DA7B2507 0160A28B 7CD1A66D
 75DF194B C217402E 7E74D466 4E39177B 81051774 25A71A0A 0F020301 0001
```

```
Key label: testuser1secondkey
Type     : RSA Encryption
Size     : 768
Created  : 16:27:54 IST Wed Sep 20 2023
Data     :
 307C300D 06092A86 4886F70D 01010105 00036B00 30680261 00B87C2F 9B4972AC
 47B40FB2 B5C10DBB 1205AD30 7E146698 2A6179AD 8F1B030D 5146C097 3A2FB3E2
 19820DA5 2132E7C7 1B7281C4 8427DF76 60E39E3A 70126DAD 108B7805 34B45915
 853956AA 301CCF4B 78F06D75 D7D90320 BE667F1D 1A479713 FD020301 0001

Key label: testuser1thirdkey
Type     : RSA General purpose
Size     : 768
Created  : 16:27:57 IST Wed Sep 20 2023
Data     :
 307C300D 06092A86 4886F70D 01010105 00036B00 30680261 00E0DDF9 53C81AE1
 35CE15E1 C7A9916F 4AED7887 65AC1E4E 48F420E4 2A56079E FD38D069 C97FC0F7
 B6D8663D C7D6FC46 1CD27EA6 AC71D36C 40E35349 0A78DA64 465B7C8B B63E8627
 BF074AF4 EC37AC0C 200AFAF3 C67E8E9B AE931964 8DF86CD9 E5020301 0001

Key label: testuser1fourthkey
Type     : RSA General purpose
Size     : 768
Created  : 16:27:57 IST Wed Sep 20 2023
Data     :
 307C300D 06092A86 4886F70D 01010105 00036B00 30680261 00E0DDF9 53C81AE1
 35CE15E1 C7A9916F 4AED7887 65AC1E4E 48F420E4 2A56079E FD38D069 C97FC0F7
 B6D8663D C7D6FC46 1CD27EA6 AC71D36C 40E35349 0A78DA64 465B7C8B B63E8627
 BF074AF4 EC37AC0C 200AFAF3 C67E8E9B AE931964 8DF86CD9 E5020301 0001
```

SSH configurations:

```
Router# show ssh
SSH version : Cisco-2.0

id       chan pty     location        state           userid           host        ver
authentication     connection type
-----------------------------------------------------------------------------------------
Incoming sessions
26       1    vty1    0/RP0/CPU0      SESSION_OPEN    testuser1     192.0.2.1       v2
 rsa-pubkey     Command-Line-Interface
27       1    vty2    0/RP0/CPU0      SESSION_OPEN    testuser1     192.0.2.2       v2
 rsa-pubkey     Command-Line-Interface
28       1    vty3    0/RP0/CPU0      SESSION_OPEN    testuser1     192.0.2.3       v2
 rsa-pubkey     Command-Line-Interface
29       1    vty4    0/RP0/CPU0      SESSION_OPEN    testuser1     192.0.2.4       v2
 rsa-pubkey     Command-Line-Interface

Outgoing sessions
1                    0/RP0/CPU0      SESSION_OPEN    testuser3  192.0.2.6          v2
  password      Command-Line-Interface
```

Delete Public Keys in the Routers

This section details different methods to delete public keys in the router:

```
Router# configure
Router(config)# crypto key zeroize authentication rsa all
Thu Sep 21 21:45:23.260 IST
Do you really want to remove all these keys ?? [yes/no]: yes
Router# commit
/* Deleting public keys for the user logged in to the router */
```

```
Router# configure
Router(config)# crypto key zeroize authentication rsa username testuser all
Thu Sep 21 21:45:23.260 IST
Do you really want to remove all these keys ?? [yes/no]: yes
Router# commit
/* Deleting public keys for any user in the router */

Router# configure
Router(config)#  no ssh server username testuser
Router# commit
/* Deleting all SSH configurations for a user in the router */

Router# configure
Router(config)# no ssh server username testuser keystring third
Router# commit
/* Deleting a specific public-key for a user using SSH configurations in the router */
```

Multi-Factor Authentication for SSH

Table 53: Feature History Table

Feature Name	Release Information	Feature Description
Multi-Factor Authentication for SSH	Release 24.1.1	You can now deploy robust authentication mechanisms for SSH connections to your routers and reduce security risks due to compromised or weak passwords. We now support multi-factor authentication (MFA)—a secure access management solution that verifies the identity of a user using multiple verification factors—for SSH login on Cisco IOS XR routers. These verification factors include a combination of login credentials such as username and password and a token, a cryptographic device, or a mobile phone with MFA application installed. No new commands or data models were introduced or modified as part of this feature.

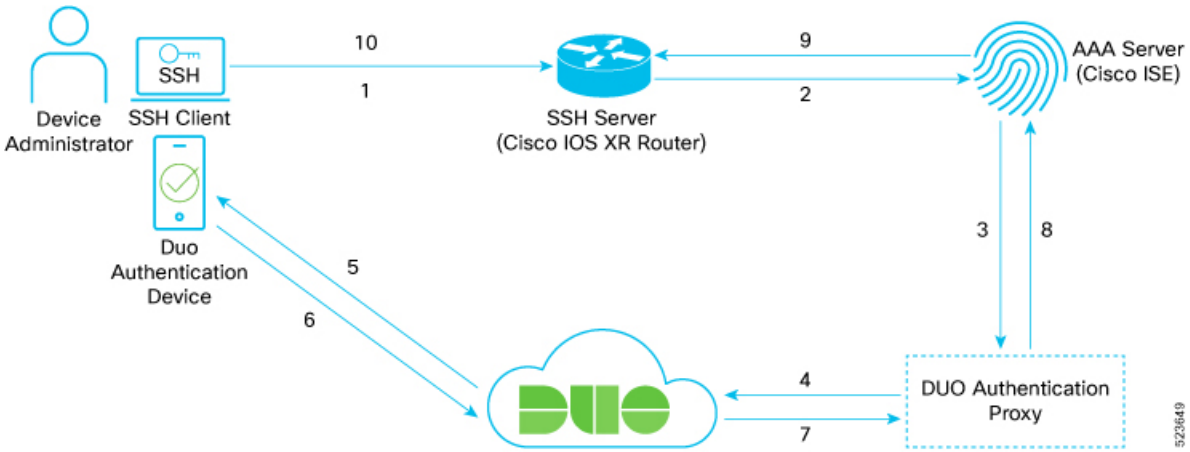
Multi-factor authentication is a multi-step authentication process that requires users to enter two or more verification factors to gain access to a system. These verification factors include something you know—such as a username and a password, and something you have—such as a token, a cryptographic authentication device, or a mobile phone with MFA application installed. MFA thereby enables stronger authentication mechanism and reduces security risk to the network devices arising due to compromised or weak passwords.

To achieve MFA for SSH, the SSH server as well as the client must support keyboard-interactive authentication method. The default order of SSH client authentication methods to support MFA in Cisco IOS XR routers is public-key, keyboard-interactive, and password-based authentication. You can change this default order as per your requirement using the **ssh client auth-method** command.

Multi-Factor Authentication Workflow

This is a sample topology to demonstrate the MFA workflow to establish SSH connection on a Cisco IOS XR router. In this example we have considered Cisco IOS XR router as the SSH server, Cisco ISE as the AAA server, and Cisco DUO authentication proxy and cloud services for MFA.

Figure 19: Multi-Factor Authentication Set-up for SSH Connection: Sample Topology



Key Components

The key components in this sample Duo MFA topology for SSH include:

- SSH client—from where the admin user initiates SSH connection to the SSH server.
- SSH server—which is the network device or router to which SSH connection is to be established.
- Cisco identity services engine (ISE)—that acts as the RADIUS or TACACS+ Server for AAA.
- DUO authentication proxy—is an on-premises software service that receives authentication requests from your local devices and applications through RADIUS or LDAP, optionally performs primary authentication against your existing LDAP directory or RADIUS authentication server, and then contacts Duo to perform secondary authentication.
- DUO cloud service—Cisco cloud-based security platform that provides secure access to any device or application.
- DUO authentication device—such as a mobile phone which has the Duo application installed.

The detailed workflow of Duo MFA for SSH is as follows:

1. The admin user initiates an SSH connection to the SSH server (Cisco IOS XR router, in this case) using the login credentials of the users that are already configured on ISE.
2. The router forwards the request to the TACACS+ AAA server (Cisco ISE, in this case).
3. The Cisco ISE sends the authentication request to Duo authentication proxy. The proxy forwards the request back to ISE for the 1st factor authentication. ISE informs the authentication proxy if the local authentication was successful.
4. Upon successful ISE authentication, the authentication proxy sends an authentication request to Duo cloud for 2nd factor authentication.

5. Duo cloud sends a *PUSH* notification to the DUO authentication device of the admin user.
6. The admin user approves the *PUSH* notification.
7. The Duo cloud informs the authentication proxy of the successful *PUSH* notification.
8. The authentication proxy informs ISE of a successful authentication.
9. The ISE authorizes the admin user.
10. The admin user successfully establishes an SSH connection with the router.

Set Up Multi-Factor Authentication for SSH

This section describes how to set up a sample topology for establishing SSH connection with Cisco IOS XR router using Duo MFA.

Prerequisites

- The Cisco IOS XR router installed with Cisco IOS XR Software Release 24.1.1 or later, that acts as the server to the SSH client, and as the client to the ISE server. The router must be already configured for AAA with ISE.
- Cisco identity services engine (ISE) server that acts as the RADIUS or TACACS+ AAA server.
- Duo MFA proxy application must be installed on either Windows or on Linux machine. For details, see https://duo.com/docs/authproxy-reference.
- DUO application must be installed on the DUO authentication device.

The procedure to set up MFA for SSH involves these high-level tasks:

- Configure Duo System
- Configure Duo Authentication Proxy
- Configure ISE
- Configure RADIUS Server Attributes on the Router
- Verify Duo MFA Set-up

Configure Duo System for MFA

Configuring Duo system for MFA involves these key steps:

1. Create a Duo account in https://duo.com/
2. Perform these Duo system configurations (for details, see the *First Steps* listed in https://duo.com/docs/radius):
   - Login to your Duo account and click on **Applications**.
   - Search for **Cisco ISE server** and click on **Protect This Application**.
   - In a notepad copy and paste your **Integration Key**, **Secret Key**, and **API Hostname**.

3. Add Duo mobile device:

   Select **Dashboard** > **Users** > ***username*** > **Add Phone**

4. Activate Duo mobile:

   Select **Dashboard** > **2FA Devices** > ***phone-number*** > **Activate Duo Mobile**

Configure Duo Authentication Proxy for MFA

Configuring Duo authentication proxy for MFA involves these key steps (For more details, see https://duo.com/docs/authproxy-reference)

1. Download and install the latest Duo authentication proxy on your Windows or Linux machine.

   In this example, we have installed the primary authentication proxy on a Windows 2016 machine and the secondary proxy on an Ubuntu server.

2. Configure the proxy for your primary authenticator.

   Edit the Duo authentication proxy configuration file, authproxy.cfg, located in the conf subdirectory of the proxy installation path in the server using a text editor. You can add multiple ISE servers as RADIUS clients and multiple router subnets/IP addresses as part of the router.

3. Start the proxy server(s) and check the proxy logs for any configuration or connectivity error.



Note For installation on Windows, ensure sure that the Windows firewall is configured to allow connections for the authentication proxy.

Configure ISE for MFA

Configuring ISE for MFA involves these key steps (for more details, see Configure Duo Two Factor Authentication for ISE Management Access)

1. Integrate ISE with Duo authentication proxy:

   a. Add a new RADIUS token server:

      Administration > **Identity Management** > **External Identity Sources** > **RADIUS Token**, and click **Add**

      Ensure that the **Shared Secret** matches the one that you already defined in the *Configure Duo Authentication Proxy* task.

      For details, see step1 listed under ISE Configuration.

   b. Set the authentication method for the identity source:

      Navigate to **Administration** > **System** > **Admin Access** > **Admin Access** > **Authentication Method**, and select previously configured RADIUS token server (for example, **RADIUS:DUO**) as the **Identity Source**.

      For details, see Step 2 listed under ISE Configuration.

2. Create device admin policies:

a. Create a policy set:

   Navigate to **Work Centers** > **Device Administration** > **Device Admin Policy Sets**.

   In this example, we created a policy set that matches on both protocols (RADIUS and TACACS+) with the **Allowed Protocols** set to **Default Device Admin**.

b. Set the following policies inside the policy set:

   - **Authentication Policy**: In this example, we have set a default rule to check the Identity Source Sequence that we defined in the steps above which contains the RADIUS Token Servers (Duo Authentication Proxies) and Active Directory.
   - **Authorization Policy**: In this example, we have set a rule that checks if the authenticated user belongs either to the **Domain Users** or **NS-ISE-IOS-Admins** groups that we have configured in active directory (AD). If the user belongs to one of these groups, then the system returns the pre-configured **Command Sets** and **Shell Profile**.

3. Add and onboard users in Duo:

   You can configure Duo to automatically sync with your AD or manually add the user in Duo (for details, see Enroll user with Duo).

Configure RADIUS Server Attributes for MFA

This topic describes how to configure RADIUS server attributes for MFA on the Cisco IOS XR router (for more details, see configure-your-radius-client(s)).

Set the IP address of the RADIUS server to the IP address of your authentication proxy, the RADIUS server port to 1812, and the RADIUS secret to the appropriate secret that you configured in the *radius_server_auto* section in the authproxy.cfg file.

```
Router#configure
Router(config)#radius-server host 209.165.200.225auth-port 1812 acct-port 1813
Router(config-radius-host)#key test@1234
Router(config-radius-host)#commit
```

Verify MFA Set-up for SSH Connection

Once you complete the Duo MFA configurations, follow these steps to verify the set-up:

- Initiate an SSH connection from the SSH client router that is already added in the ISE, using the **ssh** command.
- Use the AD credentials for the admin user to log in.
- Upon successful authentication, confirm that the user received a **Duo Push/Passcode** notification on the Duo authentication device based on what is set in the Duo authentication proxy configuration file, authproxy.cfg.
- After approving the **Duo Push** or entering the correct Passcode, the admin user must be authenticated and authorized to access the router through the SSH connection.
- The live logs of RADIUS in the ISE server must show authentication requests against the Duo authentication proxies.

• Check the authproxy log file in your authentication proxy for any errors or issues.

Selective Authentication Methods for SSH Server

Table 54: Feature History Table

Feature Name	Release Information	Feature Description
Selective Authentication Methods for SSH Server	Release 7.8.1	You now have the flexibility to choose the preferred SSH server authentication methods on the router. These methods include password authentication, keyboard-interactive authentication, and public-key authentication. This feature allows you to selectively disable these authentication methods. By allowing the SSH clients to connect to the server only through these permitted authentication methods, this functionality brings in additional security for router access through SSH. Before this release, by default, the SSH server allowed all these authentication methods for establishing SSH connections. The feature introduces these changes: • **CLI**: New **disable auth-methods** command • **YANG Data Model**: New XPaths for `Cisco-IOS-XR-crypto-ssh-cfg.yang` Cisco native model (see GitHub)

By default, the SSH server on the Cisco IOS XR routers allowed various authentication methods such as password authentication, keyboard-interactive authentication, and public-key authentication (including certificate-based authentication) for the SSH connections on the router. The SSH clients could use any of these authentication methods while attempting a connection to the SSH server on the router. From Cisco IOS XR Software Release 7.8.1, you can selectively disable these authentication methods, and allow connection attempts from the SSH client only through the remaining authentication methods. If the SSH client tries to establish a connection to the server using nonpermitted authentication methods (the ones that are disabled), then the login attempt fails.

Disable SSH Server Authentication Methods

Use the **disable auth-methods** command in ssh server configuration mode to disable the specific authentication method for the SSH server.

Public-key authentication includes certificate-based authentication as well. Hence, disabling public-key authentication automatically disables the certificate-based authentication.

Configuration Example

This example shows how to disable the keyboard-interactive authentication method for the SSH server on the router using CLI. Similarly, you can disable other authentication methods.

```
Router#configure
Router(config)# ssh server disable auth-methods keyboard-interactive
Router(config-ssh)# commit
```

Running Configuration

```
!
ssh server
 disable auth-methods keyboard-interactive
!
```

Verification

Use the **show ssh server** command to see the list of authentication methods that the SSH server on the router supports. In this example, the keyboard-interactive method is disabled and the SSH server allows all other authentication methods.

```
Router#show ssh server

Wed Feb 23 10:38:37.716 UTC
Authentication Method Supported
-----------------------------------
                 PublicKey := Yes
                  Password := Yes
      Keyboard-Interactive := No
         Certificate Based := Yes
```

SSH Port Forwarding

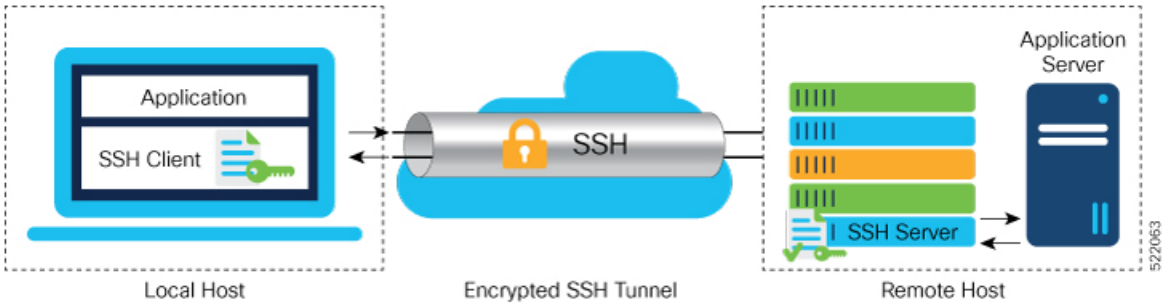
Table 55: Feature History Table

Feature Name	Release Information	Feature Description
SSH Port Forwarding	Release 7.3.2	With this feature enabled, the SSH client on a local host forwards the traffic coming on a given port to the specified host and port on a remote server, through an encrypted SSH channel. Legacy applications that do not otherwise support data encryption can leverage this functionality to ensure network security and confidentiality to the traffic that is sent to remote application servers. This feature introduces the ssh server port-forwarding local command.

SSH port forwarding is a method of forwarding the otherwise insecure TCP/IP connections from the SSH client to server through a secure SSH channel. Since the traffic is directed to flow through an encrypted SSH

connection, it is tough to snoop or intercept this traffic while in transit. This SSH tunneling provides network security and confidentiality to the data traffic, and hence legacy applications that do not otherwise support encryption can mainly benefit out of this feature. You can also use this feature to implement VPN and to access intranet services across firewalls.

Figure 20: SSH Port Forwarding

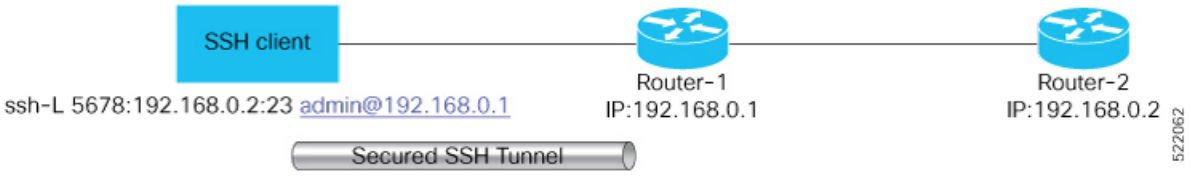


Consider an application on the SSH client residing on a local host, trying to connect to an application server residing on a remote host. With tunneling enabled, the application on the SSH client connects to a port on the local host that the SSH client listens to. The SSH client then forwards the data traffic of the application to the SSH server over an encrypted tunnel. The SSH server then connects to the actual application server that is either residing on the same router or on the same data center as the SSH server. The entire communication of the application is thus secured, without having to modify the application or the work flow of the end user.

The SSH port forwarding feature is disabled, by default. You can enable the feature by using the **ssh server port-forwarding local** command in the Global Configuration mode.

How Does SSH Port Forwarding Work?

Figure 21: Sample Topology for SSH Port Forwarding



Consider a scenario where port forwarding is enabled on the SSH server running on Router-1, in this topology. An SSH client running on a local host tries to create a secure tunnel to the SSH server, for a local application to eventually reach the remote application server running on Router-2.

The client tries to establish an SSH connection to Router-1 using the following command:

```
ssh -L local-port:remote-server-hostname:remote-port username@sshserver-hostname
```

where,

local-port is the local port number of the host where the SSH client and the application reside. Port 5678, in this example.

remote-server-hostname:remote-port is the TCP/IP host name and port number of the remote application server where the recipient (SSH server) must connect the channel from the SSH client to. 192.168.0.2 and 23, in this example.

sshserver-hostname is the domain name or IP address of the SSH server which is the recipient of the SSH client request. 192.168.0.1, in this example.

For example,

```
ssh -L 5678:192.168.0.2:23 admin@192.168.0.1
```

When the SSH server receives a TCP/IP packet from the SSH client, it accepts the packet and opens a socket to the remote server and port specified in that packet. Once the connection between SSH client and server is established, the SSH server connects that communication channel to the newly created socket. From then onwards, SSH server forwards all the incoming data from the client on that channel to that socket. This type of connection is known as port-forwarded local connection. When the client closes the connection, the SSH server closes the socket and the forwarded channel.

How to Enable SSH Port Forwarding

Guidelines for Enabling SSH Port Forwarding Feature

- The Cisco IOS XR software supports SSH port forwarding only on SSH server; not on SSH client. Hence, to utilize this feature, the SSH client running at the end host must already have the support for SSH port forwarding or tunneling.
- The application server must be reachable on the same VRF where the current SSH connection between the server and the client is established.
- Port numbers need not match for SSH port forwarding to work. You can map any port on the SSH server to any port on the client.
- If the SSH client tries to do port forwarding without the feature being enabled on the SSH server, the port forwarding fails, and displays an error message on the console. Similarly the port-forwarded channel closes in case there is any connectivity issue or if the server receives an SSH packet from the client in an improper format.

Configuration Example

```
Router#configure
Router(config)#ssh server port-forwarding local
Router(config)#commit
```

Running Configuration

```
Router#show running-configuration

ssh server port-forwarding local
!
```

Verification

Use the **show ssh** command to see the details of the SSH sessions. The **connection type** field shows as **port-forwarded-local** for the port-forwarded session.

```
Router#show ssh
```

```
Wed Oct 14 11:22:05.575 UTC
SSH version : Cisco-2.0

id chan pty  location   state        userid host          ver authentication connection
type
-----------------------------------------------------------------------------------------
Incoming sessions
15 1    XXX  0/RP0/CPU0 SESSION_OPEN  admin  192.168.122.1 v2  password
port-forwarded-local

Outgoing sessions

Router#
```

Use the **show ssh server** command to see the details of the SSH server. The **Port Forwarding** column shows as **local** for the port-forwarded session. Whereas, for a regular SSH session, the field displays as **disabled**.

```
Router#show ssh server
Tue Sep  7 17:43:22.483 IST
---------------------
SSH Server Parameters
---------------------

Current supported versions := v2
                 SSH port := 22
                 SSH vrfs := vrfname:=default(v4-acl:=, v6-acl:=)
             Netconf Port := 830
             Netconf Vrfs := vrfname:=default(v4-acl:=, v6-acl:=)

 Algorithms
---------------
        Hostkey Algorithms :=
x509v3-ssh-rsa,ecdsa-sha2-nistp521,ecdsa-sha2-nistp384,ecdsa-sha2-nistp256,rsa-sha2-512,rsa-sha2-256,ssh-rsa,ssh-dsa,ssh-ed25519

   Key-Exchange Algorithms :=
ecdh-sha2-nistp521,ecdh-sha2-nistp384,ecdh-sha2-nistp256,diffie-hellman-group14-sha1
     Encryption Algorithms :=
aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes256-gcm@openssh.com
           Mac Algorithms := hmac-sha2-512,hmac-sha2-256,hmac-sha1

 Authentication Method Supported
------------------------------------
                PublicKey := Yes
                 Password := Yes
      Keyboard-Interactive := Yes
         Certificate Based := Yes

 Others
------------
                     DSCP  := 0
                Ratelimit  := 600
            Sessionlimit  := 110
               Rekeytime  := 30
       Server rekeyvolume  := 1024
  TCP window scale factor  := 1
            Backup Server  := Disabled
          Host Trustpoint  :=
          User Trustpoint  := tes,test,x509user
          Port Forwarding  := local
Max Authentication Limit  := 16
     Certificate username  := Common name(CN) User principle name(UPN)
Router#
```

Syslogs for SSH Port Forwarding Feature

The router console displays the following syslogs at various SSH session establishment events.

- When SSH port forwarding session is successfully established:

```
RP/0/RP0/CPU0:Aug 24 13:10:15.933 IST: SSHD_[66632]:
%SECURITY-SSHD-6-PORT_FWD_INFO_GENERAL : Port Forwarding, Target:=10.105.236.155,
Port:=22, Originator:=127.0.0.1,Port:=41590, Vrf:=0x60000000, Connection forwarded
```

- If SSH client tries to establish a port forwarding session without SSH port forwarding feature being enabled on the SSH server:

```
RP/0/RP0/CPU0:Aug 24 13:20:31.572 IST: SSHD_[65883]: %SECURITY-SSHD-3-PORT_FWD_ERR_GENERAL
 : Port Forwarding, Port forwarding is not enabled
```

Associated Command

- **ssh server port-forwarding local**

Non-Default SSH Port

Table 56: Feature History Table

Feature Name	Release Information	Feature Description
Non-Default SSH Port	Release 7.7.1	We have enhanced the system security to minimize the automated attacks that may target the default Secure Socket Shell (SSH) port on your router. You can now specify a non-default port number for the SSH server on your router. The SSH, Secure Copy Protocol (SCP), and Secure File Transfer Protocol (SFTP) client services can then access your router only through this non-default port. The new port option also enables the SSH, SCP, and SFTP clients on your router to connect to SSH servers on the network that use a wide range of non-default port numbers. In earlier releases, these SSH, SCP, and SFTP connections were established through the default SSH port, 22. The non-default SSH port is supported only on SSH version 2. The feature introduces the **ssh server port** command. The feature modifies these commands to include the **port** option: • **ssh** • **sftp** • **scp**

The SSH, SCP, and SFTP services on the Cisco IOS XR routers used the default SSH port number, 22, to establish connections between the server and the client. From Cisco IOS XR Software Release 7.7.1 and later, you can specify a non-default SSH port number within a specific range for these services on Cisco IOS XR 64-bit routers. This non-default port option is available for routers that are functioning as servers, or as clients for the SSH, SCP and SFTP services. This feature helps to restrict insecure client services from accessing the router through the default SSH server port. Similarly, for Cisco IOS XR routers that are running as SSH clients, the non-default port number option enables them to connect to other SSH servers on the network that listens on a wide range of non-default SSH port numbers.

The non-default SSH port number ranges from 5520 to 5529 for the SSH server, and from 1025 to 65535 for the SSH client.

The SSH server on the router does not listen on both the default and non-default ports at the same time. If you have configured a non-default SSH server port, then the server listens only on that non-default port for the client connections. The SSH clients can then establish sessions through this non-default SSH port. The SCP and SFTP services also use the same SSH port for their connections, and hence they establish the client sessions through the newly configured port.

If a session was already established through the default port, then that session remains intact even if you change the ssh server port to a non-default port. The further client sessions are attempted through the newly configured non-default port.

Restrictions for Non-Default SSH Port

These restrictions apply to the non-default SSH port option:

- Available only on Cisco IOS XR 64-bit routers; not on 32-bit routers
- Available only on Cisco IOS XR routers that support Cisco IOS XR SSH, the classic version of SSH; not on Cisco IOS XR routers that support CiscoSSH, the OpenSSH-based implementation of SSH
- Available only on version 2 of SSH (SSHv2); not on version 1 (SSHv1)

How to Configure Non-Default SSH Port



Note To establish SSH connections on the non-default port, ensure that the non-default port that you select for the SSH server is not used by any other application on the router.

Configuration Example

SSH Server:

To configure the non-default SSH port for the SSH server on the router, use the **ssh server port** command in the Global Configuration mode.

```
Router#configure
Router(config)#ssh server port 5520
Router(config)#commit
```

SSH Client:

Similarly, the **port** option is available for the SSH client also, to initiate a connection to another SSH server that listens on a non-default SSH port number.

This example shows how to connect to an SSH server, with IP address 198.51.100.1, that is listening on non-default SSH port 5525.

```
Router#ssh 198.51.100.1 port 5525 username user1
```

Running Configuration

This is a sample running configuration of the SSH server.

```
Router#show running-configuration
!
ssh server v2
ssh server port 5520
ssh server vrf default
!
```

Verification

Use the following **show** commands to verify the SSH server configuration and LPTS entries for SSH connections.

In this example, the **SSH port** field displays the port number, '5520', that you have configured for the SSH server.

```
Router#show ssh server
Fri May 20 07:22:57.579 UTC
---------------------
SSH Server Parameters
---------------------

Current supported versions := v2
                  SSH port := 5520
                  SSH vrfs := vrfname:=default(v4-acl:=, v6-acl:=)
              Netconf Port := 830
              Netconf Vrfs :=

 Algorithms
---------------
        Hostkey Algorithms :=
x509v3-ssh-rsa,ssh-rsa-cert-v01@openssh.com,ecdsa-sha2-nistp521,ecdsa-sha2-nistp384,ecdsa-sha2-nistp256,rsa-sha2-512,rsa-sha2-256,ssh-rsa,ssh-dss,ssh-ed25519

   Key-Exchange Algorithms :=
ecdh-sha2-nistp521,ecdh-sha2-nistp384,ecdh-sha2-nistp256,diffie-hellman-group14-sha1,curve25519-sha256,diffie-hellman-group14-sha256,diffie-hellman-group16-sha512,curve25519-sha256@libssh.org

     Encryption Algorithms :=
aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes256-gcm@openssh.com,chacha20-poly1305@openssh.com

            Mac Algorithms := hmac-sha2-512,hmac-sha2-256,hmac-sha1

Authentication Method Supported
------------------------------------
                 PublicKey := Yes
                  Password := Yes
      Keyboard-Interactive := Yes
         Certificate Based := Yes

 Others
------------
                     DSCP  := 16
                Ratelimit  := 60
             Sessionlimit  := 64
                Rekeytime  := 60
       Server rekeyvolume  := 1024
  TCP window scale factor  := 1
            Backup Server  := Disabled
          Host Trustpoint  :=
          User Trustpoint  :=
```

```
          Port Forwarding  := Disabled
Max Authentication Limit  := 20
     Certificate username  := Common name(CN)
  OpenSSH Host Trustpoint  :=
  OpenSSH User Trustpoint  :=
```

In the following example, the **Port** field in the **Local-Address,Port** column for the **TCP** entry for SSH displays the port number as '5520'. This is the port on which the SSH server listens for client connections.

```
Router#show lpts bindings brief
Fri May 20 07:23:21.416 UTC

@ - Indirect binding; Sc - Scope

Location   Clnt Sc L3   L4     VRF-ID    Interface    Local-Address,Port Remote-Address,Port
---------- ---- -- ---- ------ --------- ------------ ------------------------------------
0/RP0/CPU0 IPV4 LO IPV4 ICMP   *          any          any,ECHO           any
0/RP0/CPU0 IPV4 LO IPV4 ICMP   *          any          any,TSTAMP         any
0/RP0/CPU0 IPV4 LO IPV4 ICMP   *          any          any,MASKREQ        any
0/RP0/CPU0 IPV6 LO IPV6 ICMP6  *          any          any,ECHOREQ        any
0/RP0/CPU0 IPV6 LO IPV6 ICMP6  *          any          any,NDRTRSLCT      any
0/RP0/CPU0 IPV6 LO IPV6 ICMP6  *          any          any,NDRTRADV       any
0/RP0/CPU0 IPV6 LO IPV6 ICMP6  *          any          any,NDNBRSLCT      any
0/RP0/CPU0 IPV6 LO IPV6 ICMP6  *          any          any,NDNBRADV       any
0/RP0/CPU0 IPV6 LO IPV6 ICMP6  *          any          any,NDREDIRECT     any
0/RP0/CPU0 BFD  LO IPV4 UDP    *          any          any                any
0/0/CPU0   IPV4 LO IPV4 ICMP   *          any          any,ECHO           any
0/0/CPU0   IPV4 LO IPV4 ICMP   *          any          any,TSTAMP         any
0/0/CPU0   IPV4 LO IPV4 ICMP   *          any          any,MASKREQ        any
0/0/CPU0   IPV6 LO IPV6 ICMP6  *          any          any,ECHOREQ        any
0/0/CPU0   IPV6 LO IPV6 ICMP6  *          any          any,NDRTRSLCT      any
0/0/CPU0   IPV6 LO IPV6 ICMP6  *          any          any,NDRTRADV       any
0/0/CPU0   IPV6 LO IPV6 ICMP6  *          any          any,NDNBRSLCT      any
0/0/CPU0   IPV6 LO IPV6 ICMP6  *          any          any,NDNBRADV       any
0/0/CPU0   IPV6 LO IPV6 ICMP6  *          any          any,NDREDIRECT     any
0/0/CPU0   BFD  LR IPV4 UDP    *          any          any 128.64.0.0/16
0/RP0/CPU0 TCP  LR IPV6 TCP    default    any          any,5520           any
0/RP0/CPU0 TCP  LR IPV4 TCP    default    any          any,5520           any
0/RP0/CPU0 UDP  LR IPV6 UDP    default    any          any,33433          any
0/RP0/CPU0 UDP  LR IPV4 UDP    default    any          any,33433          any
0/RP0/CPU0 RAW  LR IPV4 IGMP   default    any          any                any
0/RP0/CPU0 RAW  LR IPV4 L2TPV3 default    any          any                any
0/RP0/CPU0 RAW  LR IPV6 ICMP6  default    any          any,MLDLQUERY      any
0/RP0/CPU0 RAW  LR IPV6 ICMP6  default    any          any,LSTNRREPORT    any
0/RP0/CPU0 RAW  LR IPV6 ICMP6  default    any          any,MLDLSTNRDN     any
0/RP0/CPU0 RAW  LR IPV6 ICMP6  default    any          any,LSTNRREPORT    any

Router#
```

If the non-default port was not configured, then the SSH server listens on the default SSH port 22, and the above **Port** field displays '22'.

If a session was already established through the default port, and if you change the ssh server port to a non-default port, then the output still displays an entry for that session on the default port, 22. Another entry shows that the SSH server is listening on the newly configured non-default port. New connections establish through the non-default port, 5520, in this example.

```
Location   Clnt Sc L3   L4  VRF-ID    Interface Local-Address,Port  Remote-Address,Port
---------- ---- -- ---- --- --------- --------- -----------------  ------------------
```

```
.
.
.
0/RP0/CPU0 TCP  LR IPV4 TCP default   any       192.0.2.1,5520    198.51.100.1,37764
0/RP0/CPU0 TCP  LR IPV4 TCP default   any       any,5520 any
0/RP0/CPU0 TCP  LR IPV6 TCP default   any       any,5520 any
0/RP0/CPU0 TCP  LR IPV4 TCP default   any       192.0.2.1,22      198.51.100.1,45722
.
.
.
```

SSH server timeouts

Unused connection timeout for SSH sessions

An unused connection timeout is a secure shell (SSH) configuration setting that

- terminates SSH connections with no active channels after a specified period
- prevents accumulation of stale connections, and
- protects routers from reaching SSH session limits due to inactive sessions.

An SSH connection is a secure communication session that

- establishes an encrypted link between a client and a server
- enables authentication and data exchange, and
- serves as the foundation for creating multiple SSH channels.

Table 57: Feature History Table

Feature Name	Release Information	Feature Description
Unused connection timeout for SSH sessions	Release 25.3.1	You can prevent session limit exhaustion and maintain optimal system performance by automatically disconnecting SSH connections with no active channels. The feature introduces a configurable timeout for unused SSH connections, ensuring stale sessions do not occupy resources on your routers. The router monitors each SSH connection and terminates it when all channels remain closed and SSH clients do not create new channels within the configured timeout period. The feature introduces these changes: CLI: • **ssh server timeout** YANG Data Models: • `Cisco-IOS-XR-crypto-ssh-cfg.yang` data model was modified. • `Cisco-IOS-XR-um-ssh-cfg.yang` data model was modified. (see GitHub)

SSH connections form the basis for secure management and file transfer sessions in Cisco IOS XR routers. These connections persist after successful authentication and host one or more SSH channels for various operations. When all channels within a connection close and no new channels open, the unused connection timeout begins counting down.

The unused connection timeout functionality addresses issues where client automation scripts fail to properly close SSH connections, causing stale connections that remain active indefinitely and consume session resources.

Set unused connection timeout for SSH sessions

Follow this procedure to enable automatic disconnection of unused SSH connections on your routers to enhance device security and resource management.

Procedure

Step 1 Configure connection timeout for the unused SSH sessions on your router.

Example:

```
Router#configure
Router(config)#ssh server timeout connection 600
Router(config)#commit
```

The timeout begins counting down when all channels within a connection close and no new channels open.

Step 2 Verify the unused connection timeout configuration on the router.

Example:

```
Router#show run ssh
Thu Jul  3 09:45:41.201 UTC
ssh server v2
ssh server timeout
 connection 600
!
ssh server netconf vrf default
Router#


Router#show ssh server
Mon Aug 11 13:37:15.363 IST
---------------------
SSH Server Parameters
---------------------

Current supported versions := v2
                  SSH port := 22
                  SSH vrfs := vrfname:=default(v4-acl:=, v6-acl:=)
              Netconf Port := 830
              Netconf Vrfs := vrfname:=default(v4-acl:=, v6-acl:=)
        Netconf on SSH-port:= True (Netconf requests allowed on SSH port)

 Algorithms
---------------
        Hostkey Algorithms :=
x509v3-ssh-rsa,ssh-rsa-cert-v01@openssh.com,ssh-ed25519-cert-v01@openssh.com,ecdsa-sha2-nistp521-cert-v01@openssh.com,


ecdsa-sha2-nistp256-cert-v01@openssh.com,ecdsa-sha2-nistp256,ecdsa-sha2-nistp384,ecdsa-sha2-nistp521,ssh-ed25519,rsa-sha2-512,rsa-sha2-256,ssh-rsa,ssh-dss

   Key-Exchange Algorithms :=
ecdh-sha2-nistp521,ecdh-sha2-nistp384,ecdh-sha2-nistp256,diffie-hellman-group14-sha1,curve25519-sha256,


diffie-hellman-group14-sha256,diffie-hellman-group16-sha512,curve25519-sha256@libssh.org
     Encryption Algorithms :=
aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes256-gcm@openssh.com,chacha20-poly1305@openssh.com

            Mac Algorithms := hmac-sha2-512,hmac-sha2-256

 Authentication Method Supported
------------------------------------
                PublicKey := Yes
                 Password := Yes
      Keyboard-Interactive := Yes
         Certificate Based := Yes

 Others
------------
                     DSCP  := 48
```

```
               Ratelimit  := 6000
            Sessionlimit  := 110
               Rekeytime  := 60
      Server rekeyvolume  := 1024
 TCP window scale factor  := 1
           Backup Server  := Disabled
         Host Trustpoint  :=
         User Trustpoint  :=
         Port Forwarding  := Disabled
Max Authentication Limit  := 20
    Certificate username  := Common name(CN)
 OpenSSH Host Trustpoint  :=
 OpenSSH User Trustpoint  :=
    OpenSSH HIBA enabled  := No
       OpenSSH HIBA role  :=
            HIBA version  := 1
Punt Rx SSH pkts to NetIO := False
         Channel Timeout  := 300
      Connection Timeout  := 600
```

Step 3 Check the SSH session details on the router to confirm that the unused SSH session disconnects after the specified period of inactivity.

Example:

This sample command output shows that there is an active channel and the connection timeout has not yet triggered.

```
Router#show ssh
Mon Aug 11 08:10:39.133 UTC
SSH version : Cisco-2.0

id       chan pty     location        state           userid    host          ver authentication
connection type
-----------------------------------------------------------------------------------------------------------
Incoming sessions
2         0   vty0    0/RP0/CPU0      SESSION_OPEN    cisco     198.51.100.1  v2  password
Command-Line-Interface

Outgoing sessions
```

The connection timeout triggers when there is no active channel. The **connection type** field then displays blank.

```
Router#show ssh
Mon Aug 11 08:10:47.675 UTC
SSH version : Cisco-2.0

id       chan pty     location        state           userid    host          ver authentication
connection type
-----------------------------------------------------------------------------------------------------------
Incoming sessions
2         0   XXXXX   0/RP0/CPU0      SESSION_OPEN    cisco     198.51.100.1  v2  password


Outgoing sessions
Router#
```

This is the sample output when the connection timeout expires and entire connection closes.

```
Router#show ssh
Mon Aug 11 08:10:53.097 UTC
SSH version : Cisco-2.0

id       chan pty     location       state          userid    host                ver authentication
```

```
 connection type
----------------------------------------------------------------------------------------------------------------------------
Incoming sessions

Outgoing sessions
Router
```

Step 4 Check the system logs on the router console that indicates the termination of the inactive SSH sessions.

Example:

```
RP/0/RP0/CPU0:Aug 11 08:10:51.557 UTC: ssh_syslog_proxy[1215]: %SECURITY-SSHD_SYSLOG_PRX-6-INFO_GENERAL
 : sshd[42522]: terminating inactive connection from user user1 198.51.100.1 port 25372
```

The router automatically disconnects unused SSH connections after the configured connection timeout period.

Channel timeout for SSH sessions

A channel timeout is a secure shell (SSH) configuration setting that

- closes an SSH channel if it remains idle for a specific duration
- applies to all types of channels, including Shell, SFTP, and Netconf, and
- ensures unused channels are closed promptly.

An SSH channel is a virtual communication path within an SSH connection that

- allows concurrent operations such as shell access or file transfer
- operates independently within its parent SSH connection, and
- can be opened and closed without affecting the overall connection.

SSH channels allow multiple management or file operations to run concurrently over a single SSH connection on the router.

Table 58: Feature History Table

Feature Name	Release Information	Feature Description
Channel timeout for SSH sessions	Release 25.3.1	You can improve resource efficiency and minimize potential security risks by automatically closing idle SSH channels on the routers after a specific period of inactivity. The feature introduces a configurable timeout for SSH channels which ensures that unused channels do not persist while the parent SSH connection remains active. The router monitors each SSH channel and closes any channel where no data is sent or received within the configured timeout period. The feature introduces these changes: CLI: • **ssh server timeout** YANG Data Models: • `Cisco-IOS-XR-crypto-ssh-cfg.yang` data model was modified. • `Cisco-IOS-XR-um-ssh-cfg.yang` data model was modified. (see GitHub)

The SSH client opens one or more SSH channels such as Shell, SFTP, and so on, within the SSH connection as needed. The channel timeout begins counting down after a specific period of inactivity.

Channel timeouts help enforce resource efficiency by closing idle channels, while the parent SSH connection remains active.

Set channel timeout for SSH sessions

Follow this procedure to enable automatic disconnection of idle SSH channels belonging to active SSH sessions on your routers to enhance device security and resource management.

Procedure

Step 1 Configure channel timeout for the SSH channels on your router.

Example:

```
Router#configure
Router(config)#ssh server timeout channel 300
Router(config)#commit
```

The timer expires when the SSH channel remains idle for the specified timeout period.

Step 2 Verify the SSH channel timeout configuration on the router by running these commands.

Example:

```
Router#show run ssh
Thu Jul  3 09:45:41.201 UTC
ssh server v2
ssh server timeout
 channel 300
 connection 600
!
ssh server netconf vrf default
Router#


Router#show ssh server
```

See the sample command output here.

Step 3 Check the SSH session details on the router to confirm if the idle SSH channel closes after the specified period of inactivity.

Example:

Before timer expiry:

```
Router#show ssh
Thu Jul  3 09:46:29.807 UTC
SSH version : Cisco-2.0

id       chan pty     location        state           userid    host             ver authentication
 connection type
-----------------------------------------------------------------------------------------------------------------------
Incoming sessions
8         0   vty0    0/RP0/CPU0      SESSION_OPEN    user1     198.51.100.1     v2  password
 Command-Line-Interface

Outgoing sessions
Router#
```

After timer expiry:

```
Router#show ssh
Thu Jul  3 09:51:33.818 UTC
SSH version : Cisco-2.0

id       chan pty     location        state           userid    host             ver authentication
 connection type
-----------------------------------------------------------------------------------------------------------------------
Incoming sessions
8         0   XXXXX   0/RP0/CPU0      SESSION_OPEN    user1     198.51.100.1     v2  password


Outgoing sessions
Router#
```

Step 4 Check the system logs on the router console that indicates the closure of the idle SSH channels.

Example:

```
RP/0/RP0/CPU0:ios#RP/0/RP0/CPU0:Jul  3 09:46:32.002 UTC: ssh_syslog_proxy[1222]:
%SECURITY-SSHD_SYSLOG_PRX-6-INFO_GENERAL : sshd[39648]: Closing channel 0 of user user1 198.51.100.1
 port 34454 after 300 seconds of inactivity
```

The router automatically closes inactive SSH channels after the configured channel timeout period.

Additional References

The following sections provide references related to implementing secure shell.

Related Documents

Related Topic	Document Title
AAA commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Authentication, Authorization, and Accounting Commands on* the Cisco ASR 9000 Series Router *Software* module in *System Security Command Reference for Cisco ASR 9000 Series Routers*.
AAA configuration tasks	*Configuring AAA Services on* the Cisco ASR 9000 Series Router*Software* module in *System Security Configuration Guide for Cisco ASR 9000 Series Routers*.
Host services and applications commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Host Services and Applications Commands on* the Cisco ASR 9000 Series Router module in *IP Addresses and Services Command Reference for Cisco ASR 9000 Series Routers*.
IPSec commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples **Note** IPSec is supported only for Open Shortest Path First version 3 (OSPFv3).	*IPSec Network Security Commands on* the Cisco ASR 9000 Series Router*Software* module in *System Security Command Reference for Cisco ASR 9000 Series Routers*
SSH commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Secure Shell Commands on* the Cisco ASR 9000 Series Router *Software* module in *System Security Command Reference for Cisco ASR 9000 Series Routers*

Standards

Standards	Title
Draft-ietf-secsh-userauth-17.txt	*SSH Authentication Protocol*, July 2003
Draft-ietf-secsh-connect-17.txt	*SSH Connection Protocol*, July 2003

Standards	Title
Draft-ietf-secsh-architecture-14.txt	*SSH Protocol Architecture*, July 2003
Draft-ietf-secsh-transport-16.txt	*SSH Transport Layer Protocol*, July 2003

MIBs

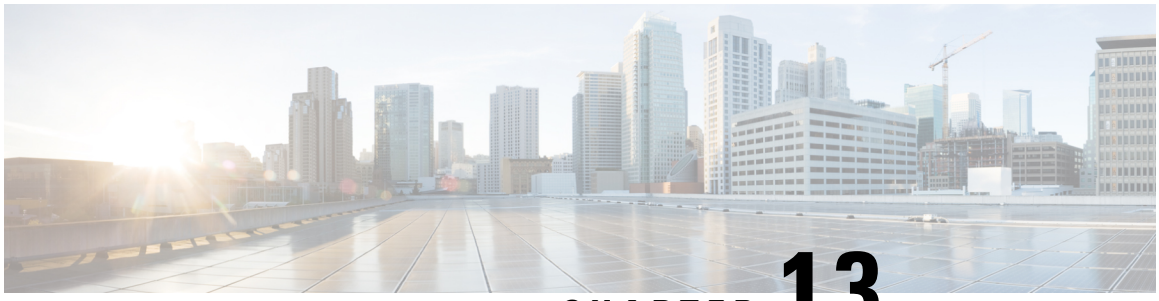
MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 6020	Netconf/ Yang

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport



CHAPTER 13

Layer 2 Security Features

This module provides an overview of security features for Layer 2 services. All Layer 2 security features must be configured at the VPLS bridge domain level.

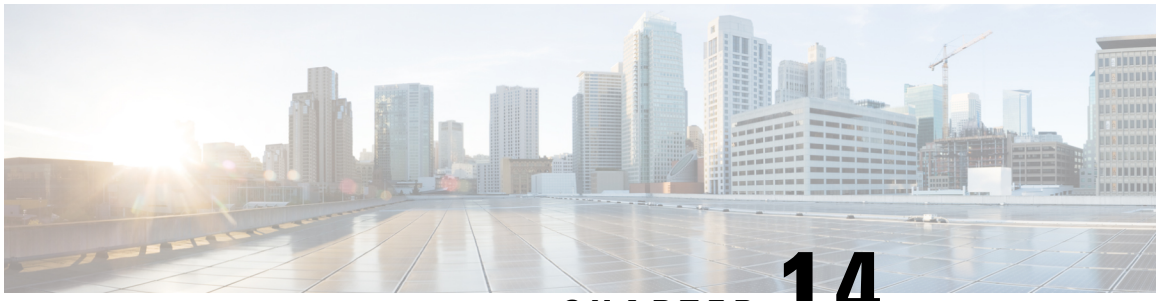
- Security Features for Layer 2 VPLS Bridge Domains, on page 431

Security Features for Layer 2 VPLS Bridge Domains

This table lists security features for Layer 2 VPLS bridge domains and points you to the detailed configuration documentation for each feature.

Table 59: Security Features for Layer 2 VPNs

Feature	Where Documented
MAC address-based traffic blocking, filtering, and limiting on VPLS bridge domains	In the *MPLS Configuration Guide for Cisco ASR 9000 Series Routers*, in the module "Implementing Virtual Private LAN Services on Cisco ASR 9000 Series Routers," see the "Configuring the MAC Address-related Parameters" section.
Traffic storm control on VPLS bridge domains	In the *System Security Configuration Guide for Cisco ASR 9000 Series Routers*(this publication), see the module "Implementing Traffic Storm Control under a VPLS Bridge on Cisco ASR 9000 Series Router."
DHCP snooping on VPLS bridge domains	In the *IP Addresses and Services Configuration Guide for Cisco ASR 9000 Series Routers*, see the module "Implementing DHCP on Cisco ASR 9000 Series Routers." That module describes both DHCP relay services and DHCP snooping at Layer 2.
IGMP snooping on VPLS bridge domains	In the *Multicast Configuration Guide for Cisco ASR 9000 Series Routers*, see the module "Implementing Layer 2 Multicast with IGMP Snooping."



CHAPTER 14

Implementing Traffic Storm Control under a VPLS Bridge

Traffic storm control provides Layer 2 port security under a Virtual Private LAN Services (VPLS) bridge by preventing excess traffic from disrupting the bridge. This module describes how to implement traffic storm control.

Feature History for Traffic Storm Control

Release	Modification
Release 3.7.2	Support was added for traffic storm control under a VPLS bridge.
Release 5.1	Support was added for: • configuring storm control at bridge domain level • allow storm control rate to be configured in kbps instead of pps

- Prerequisites for Implementing Traffic Storm Control , on page 433
- Restrictions for Implementing Traffic Storm Control, on page 434
- Information About Implementing Traffic Storm Control , on page 434
- How to Configure Traffic Storm Control, on page 436
- Configuration Examples for Traffic Storm Control , on page 442
- Additional References, on page 445

Prerequisites for Implementing Traffic Storm Control

The following prerequisites are required before implementing traffic storm control:

- The network must be configured with a VPLS bridge domain in an MPLS Layer 2 VPN.
- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Restrictions for Implementing Traffic Storm Control

The restrictions for implementing traffic storm control are as follows:

- Traffic storm control is not supported for forwarding pseudowires (VFI PWs).
- No alarms are generated when packets are dropped.
- Traffic storm control must be applied on a bridge domain, or a bridge port and not on a physical port.
- You can configure storm control on both bridge domain level and bridge port level. In this case, the storm control configured on the bridge port level will always take precedence.
- The ASR 9000 Ethernet Line Card does not support BW-based policing in kbps . However, kbps policing configuration is allowed on the ASR 9000 Ethernet Line Card. Then a conversion is performed from kbps to pps with an assumption of 1000 bytes per packet.
- In an ASR 9000 Enhanced Ethernet Line Card, the storm control is configured with mixed units. For example, broadcast policer is configured with pps and multicast policer with kbps. The policing is done in kbps, such that pps configurations are converted into kbps with an assumption of 1000 bytes per packet.

Information About Implementing Traffic Storm Control

To implement traffic storm control, you should understand the following concepts:

Understanding Traffic Storm Control

A traffic storm occurs when packets flood a VPLS bridge, creating excessive traffic and degrading network performance. Traffic storm control prevents VPLS bridge disruption by suppressing traffic when the number of packets reaches configured threshold levels. You can configure separate threshold levels for different types of traffic on each port under a VPLS bridge.

Traffic storm control monitors incoming traffic levels on a port and drops traffic when the number of packets reaches the configured threshold level during any 1-second interval. The 1-second interval is set in the hardware and is not configurable. The number of packets allowed to pass during the 1-second interval is configurable, per port, per traffic type. During this interval, it compares the traffic level with the traffic storm control level that the customer configures.

When the incoming traffic reaches the traffic storm control level configured on the bridge port, traffic storm control drops traffic until the end of storm control interval.

Traffic storm control level can be configured separately for these traffic types:

- Broadcast Traffic
- Multicast Traffic
- Unknown Unicast Traffic

The thresholds are configured using a packet-per-second (pps) and kilobit-per-second (kbps) rate. When the number of packets of the specified traffic type reaches the threshold level on a port, the port drops any additional

packets of that traffic type for the remainder of the 1-second interval. At the beginning of a new 1-second interval, traffic of the specified type is allowed to pass on the port.

Traffic storm control has little impact on router performance. Packets passing through ports are counted regardless of whether the feature is enabled. Additional counting occurs only for the drop counters, which monitor dropped packets.

No alarms are produced when packets are dropped.



Note

- Bridge Protocol Data Unit (BPDU) packets are not filtered through the storm control feature.
- Tunneled BPDU packets are filtered as they are forwarded into bridge.
- Traffic storm control is applied to only forwarded packets in the system.

Traffic Storm Control Defaults

- The traffic storm control feature is disabled by default. It must be explicitly enabled on each port for each traffic type.
- The traffic storm control monitoring interval is set in the hardware and is not configurable. On Cisco ASR 9000 Series Router, the monitoring interval is always 1 second.

Supported Traffic Types for Traffic Storm Control

On each VPLS bridge port, you can configure up to three storm control thresholds—one for each of the supported traffic types. If you do not configure a threshold for a traffic type, then traffic storm control is not enabled on that port or interface for that traffic type.

The supported traffic types are:

- Broadcast traffic—Packets with a packet destination MAC address equal to FFFF.FFFF.FFFF.
- Multicast traffic—Packets with a packet destination MAC address not equal to the broadcast address, but with the multicast bit set to 1. The multicast bit is bit 0 of the most significant byte of the MAC address.
- Unknown unicast traffic—Packets with a packet destination MAC address not yet learned.

Traffic storm control does not apply to bridge protocol data unit (BPDU) packets. All BPDU packets are processed as if traffic storm control is not configured.

Supported Ports for Traffic Storm Control

In Cisco IOS XR software Release 3.7.0 FCI, you can configure traffic storm control on the following components under a VPLS bridge domain:

- VPLS bridge domain ACs
- VPLS bridge domain access PWs

Traffic Storm Control Thresholds

Traffic storm control thresholds are configured at a packet-per-second rate. A threshold is the number of packets of the specified traffic type that can pass on a port during a 1-second interval. Valid values for traffic storm control thresholds are integers from 1 to 160000. The maximum value would permit about 19 percent of bandwidth to pass per second on a 10-Gbps link, assuming a 1500-byte packet size.

Traffic Storm Control Drop Counters

Traffic storm control counts the number of packets dropped per port and traffic type. The drop counters are cumulative until you explicitly clear them. Use the **show l2vpn bridge-domain detail** and **show l2vpn forwarding detail** commands to see drop counts. Use the **clear l2vpn forwarding counters** command to clear drop counters.

How to Configure Traffic Storm Control

This section describes how to configure traffic storm control:

Enabling Traffic Storm Control on an AC under a Bridge

Perform this task to enable traffic storm control on an AC under a VPLS bridge. The following task shows how to enable traffic storm control on an AC that is a VLAN on an Ethernet interface.



Note To disable traffic storm control, navigate to the submode you were in when you enabled the feature, and issue the **no** form of the command.

SUMMARY STEPS

1. **configure**
2. l2vpn
3. **bridge group** *bridge-group-name*
4. **bridge-domain** *bridge-domain-name*
5. **interface** *interface-name*
6. **storm-control** {**broadcast** | **multicast** | **unknown-unicast**} **pps** *packet-threshold*
7. Use the **commit** or **end** command.
8. **show l2vpn bridge-domain bd-name** *bridge-name* **detail**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	l2vpn
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters L2 VPN configuration mode.

Step 3	**bridge group** *bridge-group-name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Enters L2 VPN bridge group configuration mode.

Step 4	**bridge-domain** *bridge-domain-name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Enters L2 VPN bridge domain configuration mode.

Step 5	**interface** *interface-name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet0/1/0/0.100`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-ac)#`
		Names an AC under the bridge domain. In this case, the AC is a VLAN on an Ethernet interface.

Step 6	**storm-control** {**broadcast** | **multicast** | **unknown-unicast**} **pps** *packet-threshold*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-ac)# storm-control broadcast pps 4500`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-ac)# storm-control multicast pps 500`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-ac)#`
		Enables traffic storm control on this interface for the specified traffic type. Repeat this command for each traffic type.
		The *packet-threshold* is a packet per second rate and must be an integer between 1 and 160000. It specifies the number of packets that will be allowed to pass on the interface for the specified traffic type during a 1-second interval.

Step 7	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions:

	Command or Action	Purpose
		• **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 8	**show l2vpn bridge-domain bd-name** *bridge-name* **detail** **Example:** `RP/0/0/CPU0:router# show l2vpn bridge-domain bd-name abc detail`	Displays storm control configuration.

Enabling Traffic Storm Control on a PW under a Bridge

Perform this task to enable traffic storm control on a pseudowire under a VPLS bridge.

Note To disable traffic storm control, navigate to the submode you were in when you enabled the feature, and issue the **no** form of the command.

SUMMARY STEPS

1. **configure**
2. l2vpn
3. **bridge group** *bridge-group-name*
4. **bridge-domain** *bridge-domain-name*
5. **neighbor** *address* **pw-id** *id*
6. **storm-control** {**broadcast** | **multicast** | **unknown-unicast**} **pps** *packet-threshold*
7. Use the **commit** or **end** command.
8. **show l2vpn bridge-domain bd-name** *bridge-name* **detail**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

	Command or Action	Purpose

Step 2	l2vpn
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters L2 VPN configuration mode.

Step 3	**bridge group** *bridge-group-name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Enters L2 VPN bridge group configuration mode.

Step 4	**bridge-domain** *bridge-domain-name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Enters L2 VPN bridge domain configuration mode.

Step 5	**neighbor** *address* **pw-id** *id*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)# neighbor 10.1.1.1 pw-id 100`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-pw)#`
		Names an access pseudowire under the bridge domain.
		Note You cannot apply storm control on a for

Step 6	**storm-control** {**broadcast** | **multicast** | **unknown-unicast**} **pps** *packet-threshold*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-pw)# storm-control broadcast pps 4500`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-pw)# storm-control multicast pps 500`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-pw)#`
		Enables traffic storm control on this pseudowire for the specified traffic type. Repeat this command for each traffic type.
		The *packet-threshold* is a packet per second rate and must be an integer between 1 and 160000. It specifies the number of packets that will be allowed to pass on the interface for the specified traffic type during a 1-second interval.

Step 7	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

	Command or Action	Purpose
Step 8	**show l2vpn bridge-domain bd-name** *bridge-name* **detail** **Example:** `RP/0/0/CPU0:router# show l2vpn bridge-domain bd-name csco detail`	Displays storm control configuration settings for the named bridge domain. This command also displays the drop counter values for each configured storm control instance.

Enabling Traffic Storm Control on a Bridge Domain

Perform this task to configure traffic storm control on the bridge domain.

Note To disable traffic storm control, navigate to the submode you were in when you enabled the feature, and issue the **no** form of the command.

SUMMARY STEPS

1. **configure**
2. **l2vpn**
3. **bridge group** *bridge-group-name*
4. **bridge-domain** *bridge-domain-name*
5. **storm-control** {**broadcast** | **multicast** | **unknown-unicast**} {**kbps** | **pps**} *value*
6. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.

Step 2	**l2vpn**
	Example:
	`RP/0/0/CPU0:router(config)# l2vpn`
	`RP/0/0/CPU0:router(config-l2vpn)#`
		Enters L2 VPN configuration mode.

Step 3	**bridge group** *bridge-group-name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn)# bridge group csco`
	`RP/0/0/CPU0:router(config-l2vpn-bg)#`
		Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain..

	Command or Action	Purpose

Step 4	**bridge-domain** *bridge-domain-name*
	Example:
	`RP/0/0/CPU0:router(config-l2vpn-bg)# bridge-domain abc`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd)#`
		Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.

Step 5	**storm-control** {**broadcast** | **multicast** | **unknown-unicast**} {**kbps** | **pps**} *value*
	Example:
	`RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# storm-control multicast kbps 77`
	`RP/0/0/CPU0:router(config-l2vpn-bg-bd-pw)#`
		Configures storm control for broadcast, multicast, or unicast traffic in kilo bits per second (kbps) or as packes per second (pps).

Step 6	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Clearing Traffic Storm Control Drop Counters

Perform this task to reset traffic storm control drop counters to zero.

SUMMARY STEPS

1. clear l2vpn forwarding counters

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	clear l2vpn forwarding counters **Example:** `RP/0/0/CPU0:router# clear l2vpn forwarding counters`	Clears l2vpn forwarding counters, including storm control drop counters.

Configuration Examples for Traffic Storm Control

This section includes the following configuration examples:

Configuring Traffic Storm Control on an AC: Example

The following example shows broadcast and multicast storm control configuration on an AC under a VPLS bridge.

```
RP/0/RSP0/CPU0:router# show run

[lines deleted]

bridge group 215
  bridge-domain 215
   mtu 9000
   interface GigabitEthernet0/1/0/3.215
    storm-control multicast pps 500
    storm-control broadcast pps 4500
   !
[lines deleted]


RP/0/RSP0/CPU0:router# show l2vpn bridge-domain bd-name 215 detail
Bridge group: 215, bridge-domain: 215, id: 3, state: up, ShgId: 0, MSTi: 0
  MAC learning: enabled
  MAC withdraw: disabled
  Flooding:
    Broadcast & Multicast: enabled
    Unknown unicast: enabled
  MAC aging time: 300 s, Type: inactivity
  MAC limit: 4000, Action: none, Notification: syslog
  MAC limit reached: no
  Security: disabled
  Split Horizon Group: none
  DHCPv4 snooping: disabled
  IGMP Snooping profile: none
  Bridge MTU: 9000
  Filter MAC addresses:
  ACs: 2 (2 up), VFIs: 1, PWs: 1 (1 up)
  List of ACs:
    AC: GigabitEthernet0/1/0/3.215, state is up
      Type VLAN; Num Ranges: 1
     vlan ranges: [100, 100]
      MTU 9008; XC ID 0x440005; interworking none; MSTi 0 (unprotected)
      MAC learning: enabled
      Flooding:
        Broadcast & Multicast: enabled
        Unknown unicast: enabled
      MAC aging time: 300 s, Type: inactivity
      MAC limit: 4000, Action: none, Notification: syslog
      MAC limit reached: no
      Security: disabled
      Split Horizon Group: none
      DHCPv4 snooping: disabled
      IGMP Snooping profile: none

      Storm Control:
       Broadcast: enabled(4500)
```

```
       Multicast: enabled(500)
       Unknown unicast: disabled
      Static MAC addresses:
      Statistics:
        packet totals: receive 36728, send 31
        byte totals: receive 2791284, send 2318
        Storm control drop counters:
          packet totals: broadcast 0, multicast 0, unknown unicast 0
          byte totals: broadcast 0, multicast 0, unknown unicast 0
[lines deleted]
```

Configuring Traffic Storm Control on an Access PW: Example

The following example shows broadcast and multicast storm control configuration on an access PW under a VPLS bridge.

```
RP/0/RSP0/CPU0:router# show run
l2vpn
 bridge group bg_storm_pw
  bridge-domain bd_storm_pw
   interface Bundle-Ether101
   !
   neighbor 10.10.30.30 pw-id 1
    storm-control unknown-unicast pps 120
    storm-control multicast pps 110
    storm-control broadcast pps 100
   !
  !
 !
!

RP/0/RSP0/CPU0:router# show l2vpn bridge-domain group bg_storm_pw detail
Bridge group: bg_storm_pw, bridge-domain: bd_storm_pw, id: 2, state: up, ShgId: 0, MSTi: 0
  MAC learning: enabled
  MAC withdraw: disabled
  Flooding:
    Broadcast & Multicast: enabled
    Unknown unicast: enabled
  MAC aging time: 300 s, Type: inactivity
  MAC limit: 4000, Action: none, Notification: syslog
  MAC limit reached: no
  Security: disabled
  Split Horizon Group: none
  DHCPv4 snooping: disabled
  IGMP Snooping profile: none
  Bridge MTU: 1500
  Filter MAC addresses:
  ACs: 1 (1 up), VFIs: 0, PWs: 1 (1 up)
  List of ACs:
    AC: Bundle-Ether101, state is up
      Type Ethernet
      MTU 1500; XC ID 0xfffc0003; interworking none
      MAC learning: enabled
      Flooding:
        Broadcast & Multicast: enabled
        Unknown unicast: enabled
      MAC aging time: 300 s, Type: inactivity
      MAC limit: 4000, Action: none, Notification: syslog
      MAC limit reached: no
      Security: disabled
      Split Horizon Group: none
      DHCPv4 snooping: disabled
```

```
    IGMP Snooping profile: none
    Storm Control: disabled
    Static MAC addresses:
    Statistics:
      packets: received 0, sent 5205
      bytes: received 0, sent 645420
    Storm control drop counters:
      packets: broadcast 0, multicast 0, unknown unicast 0
      bytes: broadcast 0, multicast 0, unknown unicast 0
List of Access PWs:
  PW: neighbor 10.10.30.30, PW ID 1, state is up ( established )
    PW class not set, XC ID 0xfffc0006
    Encapsulation MPLS, protocol LDP
    PW type Ethernet, control word disabled, interworking none
    PW backup disable delay 0 sec
    Sequencing not set
  PW Status TLV in use
      MPLS         Local                          Remote
      ------------ ------------------------------ ------------------------------
      Label        16001                          16001
      Group ID     0x2                            0x2
      Interface    Access PW                      Access PW
      MTU          1500                           1500
      Control word disabled                       disabled
      PW type      Ethernet                       Ethernet
      VCCV CV type 0x2                            0x2
                   (LSP ping verification)        (LSP ping verification)
      VCCV CC type 0x6                            0x6
                   (router alert label)           (router alert label)
                   (TTL expiry)                   (TTL expiry)
      ------------ ------------------------------ ------------------------------
    Incoming Status (PW Status TLV):
      Status code: 0x0 (Up) in Notification message
    Outgoing Status (PW Status TLV):
      Status code: 0x0 (Up) in Notification message
    Create time: 16/12/2008 00:06:08 (01:00:22 ago)
    Last time status changed: 16/12/2008 00:35:02 (00:31:28 ago)
      MAC withdraw message: send 0 receive 0
    Static MAC addresses:
    Statistics:
      packets: received 0, sent 0
      bytes: received 0, sent 0
    Storm control drop counters:
      packets: broadcast 0, multicast 0, unknown unicast 0
      bytes: broadcast 0, multicast 0, unknown unicast 0
    MAC learning: enabled
    Flooding:
      Broadcast & Multicast: enabled
      Unknown unicast: enabled
    MAC aging time: 300 s, Type: inactivity
    MAC limit: 4000, Action: none, Notification: syslog
    MAC limit reached: no
    Security: disabled
    Split Horizon Group: none
    DHCPv4 snooping: disabled
    IGMP Snooping profile: none
    Storm Control:
     Broadcast: enabled(100)
     Multicast: enabled(110)
     Unknown unicast: enabled(120)
```

Configuring Traffic Storm Control on the Bridge Domain: Example

This section contains configuration examples for configuring traffic storm control on the bridge domain:

Configuring Storm Control for Broadcast Traffic: Example

This example shows how to configure storm control for broadcast traffic.

```
(config)# l2vpn
(config-l2vpn)# bridge group grp
(config-l2vpn-bg)# bridge-domain bd
(config-l2vpn-bg-bd)# storm-control broadcast kbps 770
(config-l2vpn-bg-bd)# commit
```

Configuring Storm Control for Multicast Traffic: Example

This example shows how to configure storm control for multicast traffic.

```
(config)# l2vpn
(config-l2vpn)# bridge group grp
(config-l2vpn-bg)# bridge-domain bd
(config-l2vpn-bg-bd)# storm-control multicast pps 88
(config-l2vpn-bg-bd)# commit
```

Configuring Storm Control for Unknown-Unicast Traffic: Example

This example shows how to configure storm control for unknown-unicast traffic.

```
(config)# l2vpn
(config-l2vpn)# bridge group grp
(config-l2vpn-bg)# bridge-domain bd
(config-l2vpn-bg-bd)# storm-control unknown-unicast kbps 1280
(config-l2vpn-bg-bd)# commit
```

Additional References

For additional information related to implementing traffic storm control, refer to the following references.

Related Documents

Related Topic	Document Title
MPLS Layer 2 VPNs	*Implementing MPLS Layer 2 VPNs on Cisco ASR 9000 Series Router* module in the *MPLS Configuration Guide for Cisco ASR 9000 Series Routers*.
MPLS VPLS bridges	*Implementing Virtual Private LAN Services on Cisco ASR 9000 Series Router* module in the *MPLS Configuration Guide for Cisco ASR 9000 Series Routers*
Getting started material	*Cisco ASR 9000 Series Aggregation Services Router Getting Started Guide*

Standards

Standards [1]	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

[1] Not all supported standards are listed.

MIBs

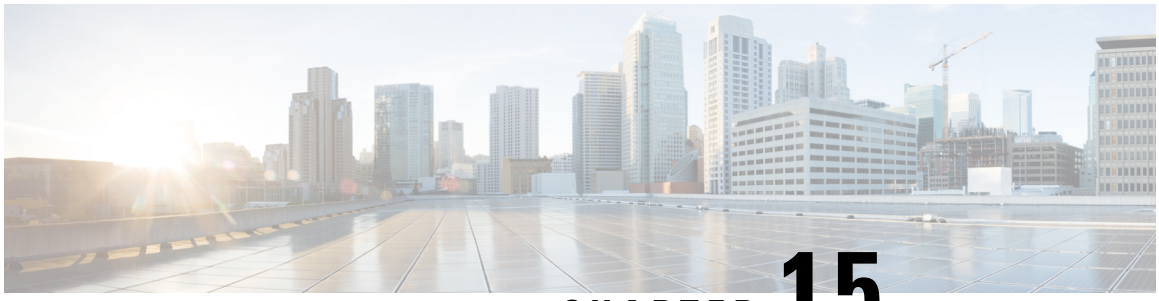
MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	—

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport



CHAPTER 15

Configuring FIPS Mode

The Federal Information Processing Standard (FIPS) 140-2 is an U.S. and Canadian government certification standard that defines requirements that the cryptographic modules must follow. The FIPS specifies best practices for implementing cryptographic algorithms, handling key material and data buffers, and working with the operating system.

In Cisco IOS XR software, these applications are verified for FIPS compliance:

- Secure Shell (SSH)
- Secure Socket Layer (SSL)
- Transport Layer Security (TLS)
- Internet Protocol Security (IPSec) for Open Shortest Path First version 3 (OSPFv3)
- Simple Network Management Protocol version 3 (SNMPv3)
- AAA Password Security

Note Any process that uses any of the following cryptographic algorithms is considered non-FIPS compliant:

- Rivest Cipher 4 (RC4)
- Message Digest (MD5)
- Keyed-Hash Message Authentication Code (HMAC) MD5
- Data Encryption Standard (DES)

The Cisco Common Cryptographic Module (C3M) provides cryptographic services to a wide range of the networking and collaboration products of Cisco. This module provides FIPS-validated cryptographic algorithms for services such as RTP, SSH, TLS, 802.1x, and so on. The C3M provides cryptographic primitives and functions for the users to develop any protocol.

By integrating with C3M, the Cisco IOS-XR software is compliant with the FIPS 140-2 standards and can operate in FIPS mode, level 1 compliance.

AAA Password Security for FIPS compliance is available from Cisco IOS XR Software Release Release 6.2.1 and later. See AAA Password Security for FIPS Compliance, on page 19.

- Prerequisites for Configuring FIPS, on page 448
- How to Configure FIPS, on page 449
- Configuration Examples for Configuring FIPS, on page 457

Prerequisites for Configuring FIPS

Install and activate the **asr9k-k9sec-px.pie** file.

Note From Cisco IOS XR Software Release 7.0.1 and later, you need not install the **asr9k-k9sec-px.pie**, because the functionality is available in the base image itself.

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command.

If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Guidelines for Enabling FIPS Mode

From Cisco IOS XR Software Release 7.1.2 and later, you must follow these guidelines while enabling FIPS mode:

- You must configure the session with a FIPS-approved cryptographic algorithm. A session configured with non-approved cryptographic algorithm for FIPS (such as, **MD5** and **HMAC-MD5**) does not work. This is applicable for OSPF, BGP, RSVP, ISIS, or any application using key chain with non-approved cryptographic algorithm, and only for FIPS mode (that is, when **crypto fips-mode** is configured).
- If you are using any **HMAC-SHA** algorithm for a session, then you must ensure that the configured *key-string* has a minimum length of 14 characters. Otherwise, the session goes down. This is applicable only for FIPS mode.
- If you try to execute the telnet configuration on a system where the FIPS mode is already enabled, then the system rejects the telnet configuration.
- If telnet configuration already exists on the system, and if FIPS mode is enabled later, then the system rejects the telnet connection. But, it does not affect the telnet configuration as such.
- It is recommended to configure the **crypto fips-mode** command first, followed by the commands related to FIPS in a separate commit. The list of commands related to FIPS with non-approved cryptographic algorithms are:
  - **key chain** *key-chain-name* **key** *key-id* **cryptographic-algorithm MD5**
  - **key chain** *key-chain-name* **key** *key-id* **cryptographic-algorithm HMAC-MD5**
  - **router ospfv3 1 authentication ipsec spi 256 md5** *md5-value*
  - **router ospfv3 1 encryption ipsec spi 256 esp des** *des-value*
  - **router ospfv3 1 encryption ipsec spi 256 esp des** *des-value* **authentication md5** *md5-value*
  - **snmp-server user** *username usergroup-name* **v3 auth md5 priv des56**

- **ssh server algorithms key-exchange diffie-hellman-group1-sha1**
- **telnet vrf default ipv4 server max-servers** *server-limit*

How to Configure FIPS

Perform these tasks to configure FIPS.

Enabling FIPS mode

SUMMARY STEPS

1. **configure**
2. **crypto fips-mode**
3. Use the **commit** or **end** command.
4. **show logging**
5. **admin**
6. **reload location all**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto fips-mode** **Example:** `RP/0/RSP0/CPU0:router(config)#crypto fips-mode`	Enters FIPS configuration mode. **Note** Stop new incoming SSH sessions while upon configuration.
Step 3	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes.

	Command or Action	Purpose
		• **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 4	**show logging** **Example:** `RP/0/RSP0/CPU0:router#show logging`	Displays the contents of logging buffers. **Note** Use the **show logging | i fips** command to fi
Step 5	**admin** **Example:** `RP/0/RSP0/CPU0:router#admin`	Enters into the admin EXEC mode.
Step 6	**reload location all** **Example:** `RP/0/RSP0/CPU0:router(admin)#reload location all`	Reloads a node or all nodes on a single chassis or multishelf system.

Configuring FIPS-compliant Keys

Perform these steps to configure the FIPS-compliant keys:

Note From Cisco IOS XR Software Release 7.0.1 and later, the crypto keys are auto-generated at the time of router boot up. Hence, you need to perform these steps to generate the keys only if the keys are missing under some scenarios.

Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 449 section for enabling the FIPS mode.

SUMMARY STEPS

1. **crypto key generate rsa [usage-keys | general-keys]** *key label*
2. **crypto key generate dsa**
3. **show crypto key mypubkey rsa**
4. **show crypto key mypubkey dsa**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**crypto key generate rsa [usage-keys | general-keys]** *key label* **Example:** `RP/0/RSP0/CPU0:router#crypto key generate rsa general-keys rsakeypair`	Generate a RSA key pair. Ensure that all the key pairs meet the FIPS requirements. The length of the key can vary from 1024 to 2048 bits. The option **usage-keys** generates separate RSA key pairs for signing and encryption. The option **general-keys** generates a general-purpose RSA key pair for signing and encryption. To delete the RSA key pair, use the **crypto key zeroize rsa** *keypair-label* command.
Step 2	**crypto key generate dsa** **Example:** `RP/0/RSP0/CPU0:router#crypto key generate dsa`	Generate a DSA key pair if required. Ensure that all the key pairs meet the FIPS requirements. The length of the key is restricted to1024 bits. To delete the DSA key pair, use the **crypto key zeroize dsa** *keypair-label* command.
Step 3	**show crypto key mypubkey rsa** **Example:** `RP/0/RSP0/CPU0:router#show crypto key mypubkey rsa`	Displays the existing RSA key pairs
Step 4	**show crypto key mypubkey dsa** **Example:** `RP/0/RSP0/CPU0:router#show crypto key mypubkey dsa`	Displays the existing DSA key pairs

Configuring FIPS-compliant Key Chain

Perform these steps to configure the FIPS-compliant key chain:

Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 449 section for enabling the FIPS mode.

SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*
4. **cryptographic-algorithm** {**HMAC-SHA1-20** | **SHA-1**}
5. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router#configure`	Enters the global configuration mode.
Step 2	**key chain** *key-chain-name* **Example:** `RP/0/RSP0/CPU0:router(config)#key chain mykeychain`	Creates a key chain.
Step 3	**key** *key-id* **Example:** `RP/0/RSP0/CPU0:router(config-mykeychain)#key 1`	Creates a key in the key chain.
Step 4	**cryptographic-algorithm** {**HMAC-SHA1-20** | **SHA-1**} **Example:** `RP/0/RSP0/CPU0:router(config-mykeychain-1)#cryptographic-algorithm HMAC-SHA1-20`	Configures the cryptographic algorithm for the key chain. Ensure that the key chain configuration always uses SHA-1 as the hash or keyed hash message authentication code (hmac) algorithm.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring FIPS-compliant Certificates

Perform these steps to configure the FIPS-compliant certificates:

Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 449 section for enabling the FIPS mode.

SUMMARY STEPS

1. **configure**

2. **crypto ca trustpoint** *ca-name key label*
3. Use the **commit** or **end** command.
4. **show crypto ca certificates**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**crypto ca trustpoint** *ca-name key label* **Example:** `RP/0/RSP0/CPU0:router(config)#crypto ca trustpoint msiox rsakeypair`	Configures the trustpoint by utilizing the desired RSA keys. Ensure that the certificates meet the FIPS requirements for key length and signature hash or encryption type. **Note** The minimum key length for RSA or DS
Step 3	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.
Step 4	**show crypto ca certificates** **Example:** `RP/0/RSP0/CPU0:router#show crypto ca certificates`	Displays the information about the certificate

Configuring FIPS-compliant OSPFv3

Perform these steps to configure the FIPS-compliant OSPFv3:

Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 449 section for enabling the FIPS mode.

SUMMARY STEPS

1. **configure**
2. **router ospfv3** *process name*
3. **area** *id*
4. **authentication** {**disable** | **ipsec spi** *spi-value* **sha1** [**clear** | **password**] *password*}
5. **exit**
6. **encryption** {**disable** | {**ipsec spi** *spi-value* **esp** {**3des** | **aes** [**192** | **256**] [**clear** | **password**] *encrypt-password*} [**authentication sha1** [**clear** | **password**] *auth-password*] } }
7. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**router ospfv3** *process name* **Example:** `RP/0/RSP0/CPU0:router(config)#router ospfv3 ospfname`	Configures the OSPFv3 process.
Step 3	**area** *id* **Example:** `RP/0/RSP0/CPU0:router(config-ospfv3)#area 1`	Configures the OSPFv3 area ID. The ID can either be a decimal value or an IP address.
Step 4	**authentication** {**disable** | **ipsec spi** *spi-value* **sha1** [**clear** | **password**] *password*} **Example:** `RP/0/RSP0/CPU0:router(config-ospfv3-ar)#authentication ipsec spi 256 sha1 password pa1`	Enables authentication for OSPFv3. Note that the OSPFv3 configuration supports only SHA-1 for authentication. **Note** IPSec is supported only for Open Shortest P
Step 5	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-ospfv3-ar)#exit`	Exits OSPFv3 area configuration and enters the OSPFv3 configuration mode.
Step 6	**encryption** {**disable** | {**ipsec spi** *spi-value* **esp** {**3des** | **aes** [**192** | **256**] [**clear** | **password**] *encrypt-password*} [**authentication sha1** [**clear** | **password**] *auth-password*] } } **Example:**	Encrypts and authenticates the OSPFv3 packets. Ensure that the OSPFv3 configuration uses the following for encryption in the configuration. • 3DES: Specifies the triple DES algorithm.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config-ospfv3)#encryption ipsec spi 256 esp 3des password pwd`	• AES: Specifies the Advanced Encryption Standard (AES) algorithm. Ensure that SHA1 is chosen if the authentication option is specified.
Step 7	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring FIPS-compliant SNMPv3 Server

Perform these steps to configure the FIPS-compliant SNMPv3 server:

Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 449 section for enabling the FIPS mode.

SUMMARY STEPS

1. **configure**
2. **snmp-server user** *username groupname* {**v3** [**auth sha** {**clear** | **encrypted**} *auth-password* [**priv** {**3des** |**aes** { **128** | **192** | **256**} } {**clear** | **encrypted** } *priv-password*]] } [**SDROwner** | **SystemOwner**] *access-list-name*
3. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router#configure`	Enters the global configuration mode.
Step 2	**snmp-server user** *username groupname* {**v3** [**auth sha** {**clear**	**encrypted**} *auth-password* [**priv** {**3des** |**aes** { **128**	**192**	**256**} } {**clear**	**encrypted**	Configures the SNMPv3 server.

	Command or Action	Purpose
	} *priv-password*]] } [**SDROwner** | **SystemOwner**] *access-list-name* **Example:** `RP/0/RSP0/CPU0:router(config)#snmp-server user user1 g v3 auth sha clear pass priv aes 128 clear privp`	
Step 3	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring FIPS-compliant SSH Client and Server

Perform these steps to configure the FIPS-compliant SSH Client and the Server:

Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 449 section for enabling the FIPS mode.

SUMMARY STEPS

1. **ssh** {*ipv4-address* | *ipv6-address*} **cipher aes** {**128-CTR** | **192-CTR** | **256-CTR**} **username** *username*
2. **configure**
3. **ssh server v2**
4. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**ssh** {*ipv4-address* | *ipv6-address*} **cipher aes** {**128-CTR** | **192-CTR** | **256-CTR**} **username** *username* **Example:** `RP/0/RSP0/CPU0:router#ssh 10.1.2.3 cipher aes 128-CTR username user1`	Configures the SSH client. Ensure that SSH client is configured only with the FIPS-approved ciphers. AES(Advanced Encryption Standard)-CTR (Counter mode) is the FIPS-compliant cipher algorithm with key lengths of 128, 192 and 256 bits.

	Command or Action	Purpose
Step 2	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 3	**ssh server v2** **Example:** `RP/0/RSP0/CPU0:router(config)#ssh server v2`	Configures the SSH server.
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuration Examples for Configuring FIPS

This section provides examples for configuring FIPS.

Configuring FIPS: Example

This example shows how to configure FIPS:

```
RP/0/3/CPU0:SSH#configure
RP/0/3/CPU0:SSH(config)#crypto fips-mode
RP/0/3/CPU0:SSH(config)#commit
RP/0/3/CPU0:SSH(config)#end
```

This example shows the output of **show logging** command:

```
RP/0/3/CPU0:SSH(config)#crypto fips-mode
RP/0/3/CPU0:SSH(config)#commit
RP/0/3/CPU0:SSH(config)#end
RP/0/3/CPU0:SSH#show logging

Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
    Console logging: level debugging, 60 messages logged
    Monitor logging: level debugging, 0 messages logged
    Trap logging: level informational, 0 messages logged
    Buffer logging: level debugging, 3 messages logged

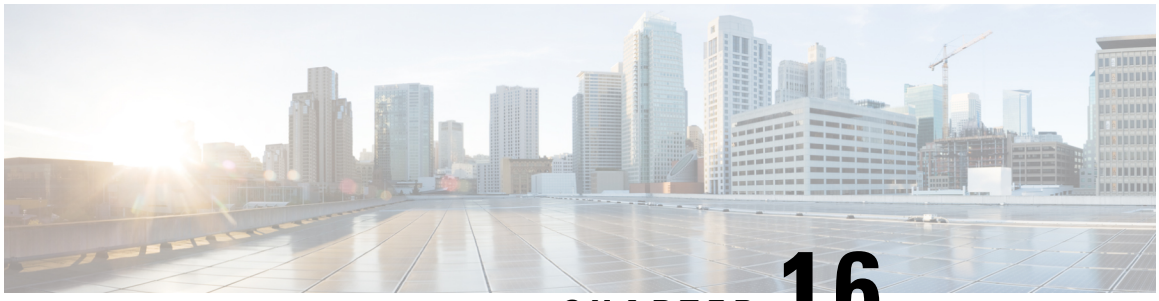
Log Buffer (9000000 bytes):
```

```
<output omitted>

Log Buffer (307200 bytes):

RP/0/RSP0/CPU0:Apr 16 12:48:17.736 : cepki[433]: The configuration setting for FIPS mode
has been modified. The system must be reloaded to finalize this configuration change. Please
 refer to the IOS XR System Security Configuration Guide, Federal Information Process
Standard(FIPS) Overview section when modifying the FIPS mode setting.
RP/0/RSP0/CPU0:Apr 16 12:48:17.951 : config[65757]: %MGBL-CONFIG-6-DB_COMMIT :
Configuration committed by user 'lab'. Use 'show configuration commit changes 1000000002'
to view the changes.
RP/0/RSP0/CPU0:Apr 16 12:48:23.988 : config[65757]: %MGBL-SYS-5-CONFIG_I : Configured from
 console by lab


....
....
....
```



CHAPTER 16

Implementing Cisco ASR 9000 vDDoS Mitigation

This module provides information about how to implement Cisco ASR 9000 vDDoS mitigation to protect network infrastructures and resources from distributed denial-of-service (DDoS) attacks.

- Cisco ASR 9000 vDDoS Mitigation Overview, on page 459
- Information about Implementing Cisco ASR 9000 vDDoS Mitigation, on page 460

Cisco ASR 9000 vDDoS Mitigation Overview

Distributed denial-of-service (DDoS) attacks target network infrastructures or computer services resources. The primary goal of DDoS attacks is to deny legitimate users access to a particular computer or network resources, which results in service degradation, loss of reputation, and irretrievable data loss. DDoS Mitigation is the process of detecting increasingly complex and deceptive assaults and mitigating the effects of the attack to ensure business continuity and resource availability.

The Arbor Peakflow solution protects customer networks by mitigating undesirable traffic caused by DDoS attacks. It comprises a number of functions as well as a set of hardware devices that implement those functions. Peakflow SP means the control components such as monitoring the network, detecting attacks, and coordinating an attack response. Peakflow SP runs on SP appliances or in virtual machines. Peakflow Threat Management System (TMS) or Peakflow SP TMS is the data plane component to remove DDoS attacks.

Using Netflow and BGP, Arbour Peakflow solution monitors the network ingress points to build a base line for network behavior and traffic patterns. It will then perform ongoing monitoring to detect anomalies and flag them as potential attacks. These potential attacks are presented to network operations via a GUI, email, or SNMP which allows a range of actions to be taken, including initiating a response or marking an event as a false alarm. If there is an attack, the Arbor Peakflow solution redirects all traffic for the destination through the TMS which can remove unwanted traffic and clean the traffic as effectively as possible without blocking valid connections. The new path to the TMS where the traffic from the original path is diverted is called off-ramp traffic path. The path from the TMS egress interface to the original destination of the traffic where the clean traffic is sent is called on-ramp traffic path.

Cisco has partnered with Arbor Networks to deliver DDoS attack mitigation capabilities on Cisco ASR 9000 Series routers by integrating the Threat Management System (TMS) DDoS mitigation functionality to the Cisco ASR 9000 router. The TMS will be implemented on the ASR 9000 VSM (Virtualized Services Module) hosted in the ASR 9000 chassis.

Information about Implementing Cisco ASR 9000 vDDoS Mitigation

There are different ways to implement DDoS mitigation. In the centralized model, a dedicated part of the network will be the scrubbing center (TMS) to clean the traffic and the traffic to the victim will be diverted to the scrubbing center. In the distributed approach, scrubbers are installed at the edge of the network. In the mixed approach, scrubbers will be present at the edge and the scrubbing center will handle the additional traffic. You should choose the mitigation strategy suitable for your network.

The mechanisms to create an effective diversion and re-injection path include BGP Flowspec, injecting a more specific route by diverting traffic to the victim in to the TMS, tunneling traffic to the TMS and from the TMS, putting the malicious and clean traffic in different VRFs or VPNs, and using ACL Based Forwarding (ABF) to steer traffic. These tools can be used in different combinations like tunnel diversion & VRF re-injection, diversion using a /32 prefix and VPN re-injection, and /32 diversion and GRE tunnel re-injection to implement a range of routing designs.

Prerequisites for Implementing Cisco ASR 9000 vDDoS Mitigation

These prerequisites are required to implement DDoS Mitigation support on the Cisco ASR 9000 Series Router.

- You need Cisco IOS XR software release 5.3.0 or later installed on the Cisco ASR 9000 Series Router.
- ASR 9000 Series Route Switch Processor 440 (RSP 440) or above is required.
- You need to insert the VSM card in the Cisco ASR 9000 Series Router.
- TFTP should be enabled on the CIsco ASR 9000 Series Router.
- You need to uninstall any pre-existing virtual service on the VSM card.
- You need to pair the ASR 9000 vDDoS solution with Arbor Peakflow SP.

Restrictions for Implementing Cisco ASR 9000 vDDoS Mitigation

The following restriction apply for implementing Cisco ASR 9000 vDDoS mitigation.

- Only one vDDoS instance is supported per VSM card.

Configuring Cisco ASR 9000 vDDoS Mitigation

This section provides information about the configuration tasks required for implementing ASR 9000 vDDoS mitigation. This section only provides information about Cisco ASR9000 specific configuration. For Arbor Peakflow SP configuration, see *Arbor Networks SP and Threat Management System (TMS) User Guide* .

Installing Cisco ASR 9000 vDDoS Software

Arbor Networks TMS and ArbOS are packaged together with configuration files in an Open Virtualization Archive (.ova) file. Installation of ASR 9000 vDDoS software on the VSM card consists of the following steps:

1. Copy the OVA file that contains Arbor TMS and Arbor OS to the ASR 9000 router using TFTP or FTP. Use the correct path and filename for your build. When you are prompted for the remote host, type the IP address of the remote host. For destination file name, press enter.

```
RP/0/RSP0/CPU0:router# copy tftp:/Peakflow-TMS-8.0.0-EKU0.ova disk0:
```

2. Enable the virtual service.

```
RP/0/RSP0/CPU0:router# virtual-service enable
RP/0/RSP0/CPU0:router# commit
```

3. Install the TMS VSM software.

```
RP/0/RSP0/CPU0:router# virtual-service install name tms3 package
/disk0:/Peakflow-TMS-8.0.0-EKU0.ova node 0/1/CPU0
```

   The installation may take 10-12 minutes to complete.

4. Check the progress of the installation process by using the **show virtual-service list** command.

```
RP/0/RSP0/CPU0:router# show virtual-service list
```

   If installation is in process, this command shows the status as installing. When installation is complete, you can rerun this show command to verify that the virtual service is listed as installed.

Configuring Interfaces for TMS Mitigation

Once you install the VSM module, twelve virtual Network Interface Card (vNIC) interfaces are available between the VSM module and the router. You can use some of these vNIC interfaces for TMS mitigation and others for management of the TMS virtual instance. The mitigation interfaces are bundled into a single logical interface. The logical interface can be divided into subinterfaces for diversion and re-injection of traffic.

1. Map vNIC interfaces on the router to TMS interfaces on the VSM card.

```
RP/0/RSP0/CPU0:router(config)# virtual-service tms3
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/0
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/1
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/2
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/3
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/4
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/5
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/6
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/7
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/8
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/9
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/10
RP/0/RSP0/CPU0:router(config-virt-service)# vnic interface tenGigE 0/1/1/11
RP/0/RSP0/CPU0:router(config-virt-service)# commit
RP/0/RSP0/CPU0:router(config-virt-service)# activate
RP/0/RSP0/CPU0:router(config-virt-service)# commit
```

2. Check the progress of the activation process by using the **show virtual-service list** command.

```
RP/0/RSP0/CPU0:router# show virtual-service list
```

   Once the VM is activated, the status changes to activated.

3. Create ethernet bundle interface for mitigation interfaces 0-3 and 7-10.

```
RP/0/RSP0/CPU0:router(config)# interface Bundle-Ether  2
RP/0/RSP0/CPU0:router(config-if)# ipv4 address 10.1.2.3 255.0.0.0
RP/0/RSP0/CPU0:router(config-if)# bundle load-balancing hash src-ip
RP/0/RSP0/CPU0:router(config-if)# exit
```

4. Add member interfaces to the ethernet bundle.

```
RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/0
RP/0/RSP0/CPU0:router(config-if)# bundle-id 2 mode on
RP/0/RSP0/CPU0:router(config-if)# load interval 30
RP/0/RSP0/CPU0:router(config-if)# commit

RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/1
RP/0/RSP0/CPU0:router(config-if)# bundle-id 2 mode on
RP/0/RSP0/CPU0:router(config-if)# load interval 30
RP/0/RSP0/CPU0:router(config-if)# commit

RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/2
RP/0/RSP0/CPU0:router(config-if)# bundle-id 2 mode on
RP/0/RSP0/CPU0:router(config-if)# load interval 30
RP/0/RSP0/CPU0:router(config-if)# commit

RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/3
RP/0/RSP0/CPU0:router(config-if)# bundle-id 2 mode on
RP/0/RSP0/CPU0:router(config-if)# load interval 30
RP/0/RSP0/CPU0:router(config-if)# commit


RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/7
RP/0/RSP0/CPU0:router(config-if)# bundle-id 2 mode on
RP/0/RSP0/CPU0:router(config-if)# load interval 30
RP/0/RSP0/CPU0:router(config-if)# commit

RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/8
RP/0/RSP0/CPU0:router(config-if)# bundle-id 2 mode on
RP/0/RSP0/CPU0:router(config-if)# load interval 30
RP/0/RSP0/CPU0:router(config-if)# commit

RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/9
RP/0/RSP0/CPU0:router(config-if)# bundle-id 2 mode on
RP/0/RSP0/CPU0:router(config-if)# load interval 30
RP/0/RSP0/CPU0:router(config-if)# commit

RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/10
RP/0/RSP0/CPU0:router(config-if)# bundle-id 2 mode on
RP/0/RSP0/CPU0:router(config-if)# load interval 30
RP/0/RSP0/CPU0:router(config-if)# commit
```

5. Configure TMS management interfaces 5 and 6.

```
RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/5
RP/0/RSP0/CPU0:router(config-if)# ip address 10.2.1.10 255.255.255.0
RP/0/RSP0/CPU0:router(config-if)# load interval 30
RP/0/RSP0/CPU0:router(config-if)# commit
RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/6
RP/0/RSP0/CPU0:router(config-if)# ip address 10.2.1.5 255.255.255.0
RP/0/RSP0/CPU0:router(config-if)# load interval 30
RP/0/RSP0/CPU0:router(config-if)# commit
```

6. Set up unused interfaces 4 and 11.

```
RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/4
RP/0/RSP0/CPU0:router(config-if)# shut down
RP/0/RSP0/CPU0:router(config-if)# commit
RP/0/RSP0/CPU0:router(config)# interface TenGigE0/1/1/11
RP/0/RSP0/CPU0:router(config-if)# shut down
RP/0/RSP0/CPU0:router(config-if)# commit
```

7. Configure subinterfaces for diversion and re-injection.

```
RP/0/RSP0/CPU0:router(config)# interface Bundle-Ether 2.100
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 100
RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 10.1.2.100 255.255.255.240
RP/0/RSP0/CPU0:router(config-subif)#  bundle load-balancing hash src-ip
RP/0/RSP0/CPU0:router(config)# interface Bundle-Ether 2.101
RP/0/RSP0/CPU0:router(config-subif)# encapsulation dot1q 101
RP/0/RSP0/CPU0:router(config-subif)# vrf onramp
RP/0/RSP0/CPU0:router(config-subif)# ipv4 address 10.1.2.130 255.255.255.240
```

Uninstalling the TMS Virtual Service

Before installing the TMS software on VSM card, you need to remove any existing TMS virtual service on the VSM card. Perform the followings steps to remove any instances of the TMS virtual service.

1. Enable the virtual services on the VSM card.

   ```
   RP/0/RSP0/CPU0:router(config)# virtual-service enable
   RP/0/RSP0/CPU0:router(config)# commit
   ```

2. Use the **show virtual-service list** command to see the list of virtual services available on the VSM card.

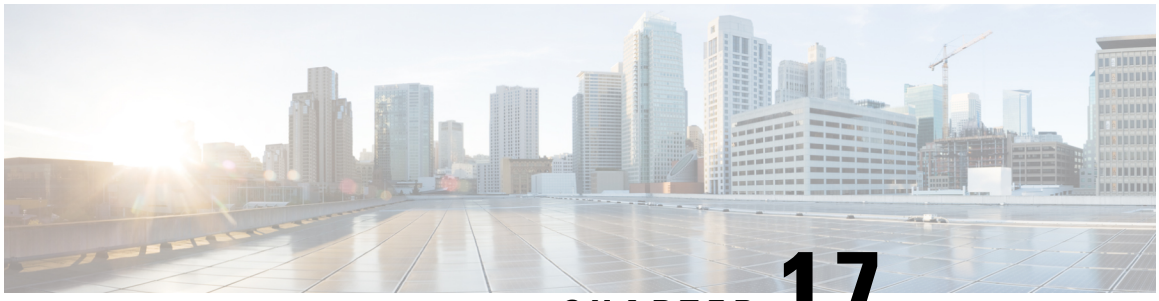
   ```
   RP/0/RSP0/CPU0:router# show virtual-service list
   ```

3. If the TMS virtual instance is listed, de-activate the TMS virtual instance.

   ```
   RP/0/RSP0/CPU0:router(config)# no virtual-service tms3
   RP/0/RSP0/CPU0:router(config)# commit
   ```

4. Uninstall the TMS virtual instance.

   ```
   RP/0/RSP0/CPU0:router# virtual-service uninstall name tms3 node 0/1/CPU0
   ```



CHAPTER 17

Implementing Secure Logging

This chapter describes the implementation of secure logging on the Cisco ASR 9000 Series Routers over Transport Layer Security (TLS). TLS, the successor of Secure Socket Layer (SSL), is an encryption protocol designed for data security over networks.

Table 60: Feature History Table

Release	Modification
Release 6.2.1	This feature was introduced.

- System Logging over Transport Layer Security (TLS), on page 465
- Restrictions for Syslogs over TLS, on page 467
- Configuring Syslogs over TLS, on page 467
- Security template framework for TLS applications, on page 469
- TLS RFC 5289 compliance for security template, on page 478

System Logging over Transport Layer Security (TLS)

System Log (syslog) messages indicate the health of the device and provide valuable information about any problems encountered. By default, the syslog process sends messages to the console terminal.

Due to limited size of the logging buffer in a router, these syslog messages get overwritten in a short time. Moreover, the logging buffer doesn't retain syslogs across router reboots. To avoid these issues, you can configure the router to send syslog messages to an external syslog server for storage.

Note For more information on configuring system logging, see *Implementing Logging Services* chapter in the *System Monitoring Configuration Guide for Cisco ASR 9000 Series Routers.*

Traditionally, routers transfer syslogs to an external syslog server using User Datagram Protocol (UDP), which is an insecure way of transferring logs. To guarantee secure transport of syslogs, Cisco ASR 9000 Series Router supports Secure Logging based on RFC 5425 (Transport Layer Security Transport Mapping for Syslog). With this feature, the router sends syslogs to a remote server, over a trusted channel which implements the secure Transport Layer Security (TLS) encryption protocol.

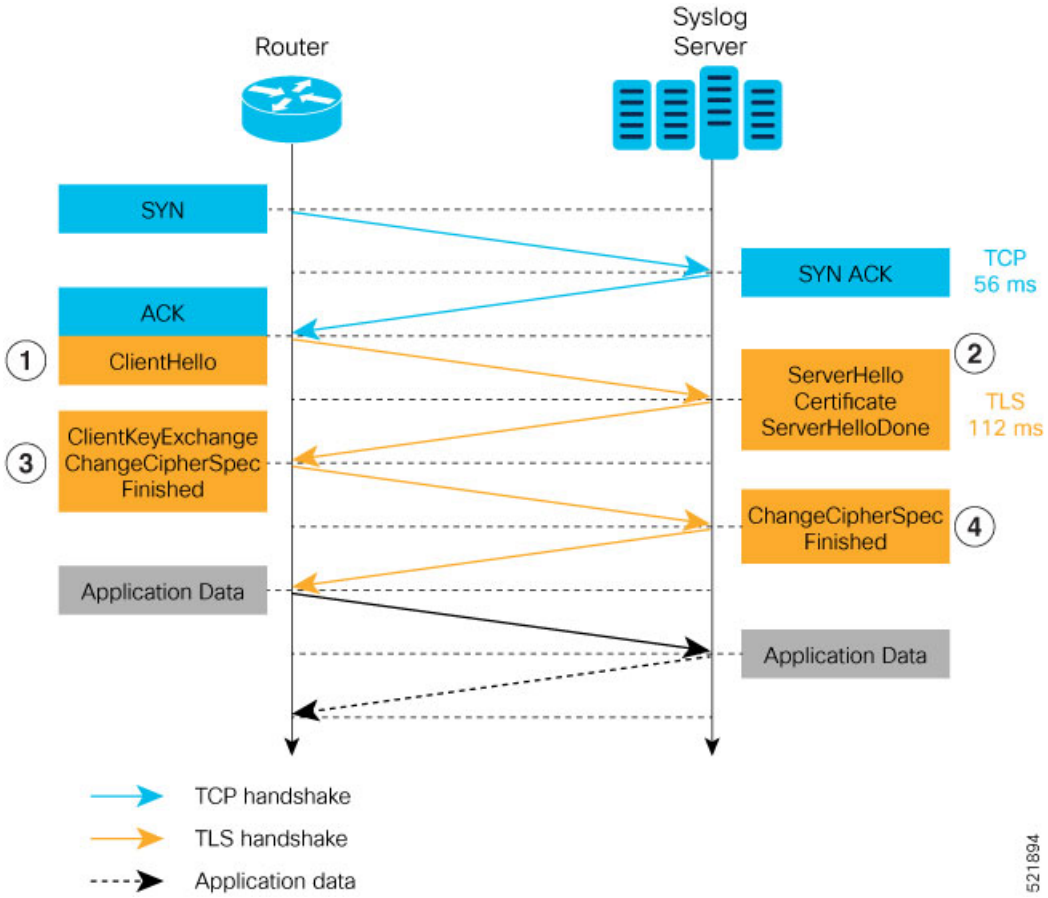
TLS ensures secure transport of syslogs by:

- Authenticating the server and client
- Encrypting the syslog data transferred
- Verifying the integrity of data

The Cisco ASR 9000 Series Router is the TLS client and remote syslog server is the TLS server. TLS runs over Transmission Control Protocol (TCP). So, the client must complete the TCP handshake with the server before starting TLS handshake.

Sequence of TLS Handshake

Figure 22: TLS Handshake



To establish the TLS session, the following interactions take place between the router and the syslog server after TCP handshake is complete:

1. The router sends Client Hello message to the server to begin TLS handshake.
2. The server shares its TLS certificate, which contains its public key and a unique session key, with the router to establish a secure connection. Each TLS certificate consists of a key pair made of a public key and private key.

3. The router confirms the server certificate with the Certification Authority and checks the validity of the TLS certificate. Then, the router sends a Change Cipher Spec message to the server to indicate that messages sent are encrypted using the negotiated key and algorithm.
4. The server decrypts the message using its private key. And then, sends back a Change Cipher Spec message encrypted with the session key to complete the TLS handshake and establish the session.

For more information on configuring Certification Authority interoperability, refer *Implementing Certification Authority Interoperability* chapter in this guide.

Restrictions for Syslogs over TLS

The following restrictions apply for sending syslogs to a remote syslog server over TLS:

- When you configure the remote syslog server on the router, specify only one identifier for a remote syslog server: its hostname or its IPv4 or IPv6 address. A single server identifier gives the TLS configuration one unambiguous endpoint identity.

Configuring Syslogs over TLS

The following steps show how to configure syslog over TLS:

1. Configure the trust-point for establishing the TLS channel as shown:

```
Router#conf t
Router(config)#crypto ca trustpoint tp
Router(config-trustp)#subject-name CN=new
Router(config-trustp)#enrollment terminal
Router(config-trustp)#rsakeypair k1
Router(config-trustp)#commit
```

Note You can either use the command **enrollment url** *SCEP-url* or the command **enrollment terminal** for configuring trustpoint certification authority (CA) enrollment. For more information, see *Implementing Certification Authority Interoperability* chapter in this guide.

2. Configure the settings to access the remote syslog server. You can use either the IPv4/v6 address of the server or the server hostname for this configuration. Based on the configured **severity**, the router sends syslogs to the server. Logging severity options include **alerts, critical, debugging, emergencies, errors, informational, notifications and warnings**. For more information about logging severity levels, see *Syslog Message Severity Level Definitions* topic in *Implementing Logging Services* chapter in *System Monitoring Configuration Guide for Cisco ASR 9000 Series Routers*.

   This example shows you how to configure syslog server settings with the IPv4 address.

```
Router(config)#logging tls-server TEST
Router(config-logging-tls-peer)#severity debugging
Router(config-logging-tls-peer)#trustpoint tp
Router(config-logging-tls-peer)#address ipv4 10.105.230.83
Router(config-logging-tls-peer)#commit
```

Alternately, you can configure the syslog server settings with server hostname instead of the IPv4/v6 address.

```
Router(config)#logging tls-server TEST
Router(config-logging-tls-peer)#severity debugging
Router(config-logging-tls-peer)#trustpoint tp
Router(config-logging-tls-peer)#tls-hostname xyz.cisco.com
Router(config-logging-tls-peer)#commit
```

3. Configure the domain to map the IP address of the remote syslog server and its hostname.

```
Router(config)#domain ipv4 host xyz.cisco.com 10.105.230.83
Router(config)#domain name cisco.com
Router(config)#commit
```

Verification Steps

TCP port 6514 is the default port for syslog over TLS. Verify the TLS configuration by checking if port 6514 is associated with the IP address of the syslog server in the output of the command **show lpts bindings brief**.

```
Router#show lpts bindings brief

@ - Indirect binding; Sc - Scope

 Location   Clnt Sc L3   L4    VRF-ID    Interface    Local-Address,Port Remote-Address,Port

 ---------- ---- -- ---- ------ --------- ------------ ---------------------------------------

  0/RP0/CPU0 TCP LR IPV4 TCP    default   any          5.10.18.5,35926 10.105.230.83,6514
```

The output of **show logging** command displays the IP address of the TLS server and the number of messages sent to the remote syslog server.

```
Router#show logging

Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
    Console logging: level debugging, 185 messages logged
    Monitor logging: level debugging, 94 messages logged
    Trap logging: level informational, 0 messages logged
    Logging to TLS server 10.105.230.83, 66 message lines logged
    Buffer logging: level debugging, 183 messages logged

Log Buffer (2097152 bytes):
...................
```

The output of **show crypto ca certificates** command displays the Certification Authority (CA) certificate details.

```
Router#show crypto ca certificates

Trustpoint       : tp
==================================================
CA certificate
  Serial Number  : B5:68:C8:96:A4:7C:1A:BA
  Subject:
      CN=cacert,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Issued By      :
      CN=cacert,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Validity Start : 05:39:51 UTC Tue Aug 13 2019
  Validity End   : 05:39:51 UTC Mon Aug 08 2039
```

```
  CRL Distribution Point
      http://10.105.236.78/crl_xxx/crl.der
  SHA1 Fingerprint:
      03BD57E04A2AA4648A84F515A46EF99CCF488387
```

When the TLS channel between the router and syslog server comes up, the router displays the following syslog messages on the console:

```
RP/0/RP0/CPU0: syslogd[148]: %SECURITY-XR_SSL-6-CERT_VERIFY_INFO : SSL Certificate
verification: Peer certificate verified successfully
RP/0/RP0/CPU0: syslogd[148]: %OS-SYSLOG-5-LOG_NOTICE : Secure Logging: Successfully
established TLS session , server :10.105.230.83
```

Security template framework for TLS applications

A security template is a configuration bundle that:

- centralizes and standardizes security policy configuration for TLS-enabled applications,
- encapsulates certificate authentication policies, TLS protocol controls, and compliance mode settings, and
- acts as a reusable, named source of truth that multiple applications can reference instead of embedding security settings locally.

Security templates simplify management by letting applications reference a named template, avoiding duplicated or inconsistent security settings.

Table 61: Feature History Table

Feature Name	Release Information	Feature Description
Security template framework for TLS enabled applications	Release 25.4.1	Security templates reduce misconfiguration risks and operational overhead by centralizing and standardizing security policy configuration for TLS-enabled applications. A security template bundles certificate authentication policy, TLS controls, and compliance mode settings. It acts as a single source of truth that applications reference, avoiding local embedding of security settings. This template defines how certificates are handled and controls various aspects of the TLS handshake. The feature introduces these changes: CLI: • **security-template**

The security template infrastructure centralizes and standardizes the configuration of certificate-based authentication and TLS settings for applications. It provides a reusable, named configuration bundle—called a security template—which can be referenced by multiple applications to enforce consistent security policies and reduce operational overhead.

Security templates reduce operational overhead and misconfiguration risk by enabling administrators to manage security as reusable templates. They support features such as:

- Version and cipher suite control for TLS,
- Selection of elliptic curve groups and digital signature algorithms,
- Integration with certificate onboarding methods such as SCEP, file-based, and Certz profiles,
- Change notification and automatic configuration updates for registered applications.

Key Terminologies

- **Certificate authentication policy:** A set of rules specifying how certificates are used for authenticating servers or clients, including trust anchor selection and certificate validation settings.
- **Common Criteria (CC) mode:** An enhanced security mode that enforces stricter compliance-focused behavior.
- **Certz profile:** A profile providing identity certificates, private keys, and CA bundles from a centralized certificate management service. For more information, see github.

Benefits of Security template framework

Security template framework offers these benefits:

- Centralized Security Management: It allows you to create and manage security templates that apply to multiple applications.
- Flexibility: The template supports configurations like Common Criteria (CC) mode and certificate authentication policies. It also allows for future expansion with additional security policies.
- Simplified Configuration: Centralized security templates reduce the complexity of managing application-specific settings.
- Consistency and Compliance: The uniform application of policies ensures adherence to organizational and regulatory standards.
- Extensibility: The system easily adapts to future security needs. It accommodates additional fields and configurations.
- Operational Efficiency: Automated notifications minimize errors and enable faster updates.

Use cases

Examples of use cases for the security template framework include:

- Managing TLS settings for Syslog.
- Supporting web-scale environments with numerous root CAs and custom certificate validation needs.
- Enforcing advanced crypto policies per application. For example, specific elliptic curves and signature algorithms.

Restriction for security template

Currently, only syslog messages sent over TLS can utilize security templates.

Configuration guidelines for security template framework

Consider these best practices and guidelines to ensure scalable, consistent, and secure management of TLS and certificate configurations for all supported applications.

- Ensure that you centralize security policy configuration for TLS-enabled applications using security templates. This standardizes certificate authentication, TLS controls, and compliance settings across services, reducing operational overhead and misconfiguration risk.
- Use a unique name for each security template and apply it as the single source of truth for applications, instead of embedding security settings locally.
- Enable Common Criteria (CC) mode for enhanced compliance where required.
- Configure certificate authentication policies to specify identity certificates and peer trust anchors as required. Prefer sourcing certificates from a Certz profile when available, as it takes precedence over individual trustpoints.

- When both a standalone trustpoint and a security template are configured under syslog server, the trustpoint specified within the security template overrides the standalone trustpoint.
- Register applications to receive and handle security template update notifications so that changes are applied dynamically and consistently.

How security template framework works

As organizations scale their network infrastructure and deploy more TLS-enabled applications, managing security settings like certificate authorities, protocol versions, cipher suites, and compliance requirements becomes complex. A centralized security template framework addresses operational inefficiencies, compliance challenges, and risk of misconfiguration by providing reusable, consistent policy bundles.

Summary

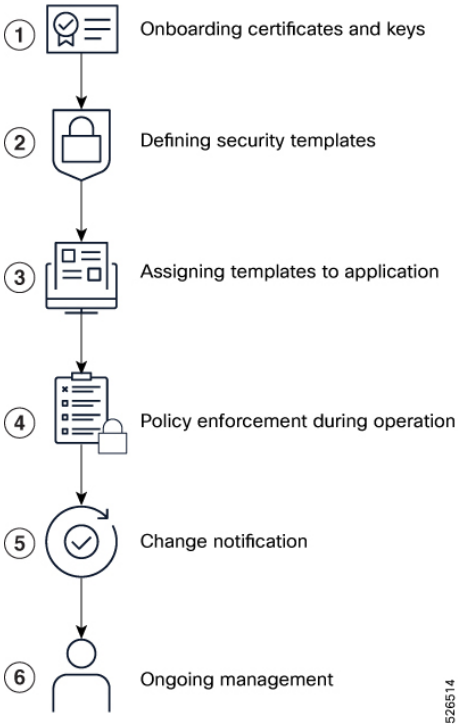
Security templates in Cisco IOS XR centralize and standardize the configuration and enforcement of TLS and certificate-based security settings across multiple applications. This approach streamlines management, reduces misconfiguration risk, and ensures consistent policy application.

The key components involved in the process are:

- **Security template infrastructure:** Central repository and logic that manages security templates and their application to different XR services.
- **Applications like Syslog:** Consume and enforce security settings by referencing named security templates.
- **Trustpoints and Certz profiles:** Provide certificate and key material for authentication and trust validation.
- **Notification and change management mechanism:** Ensures applications are updated when security templates change.

Workflow

Figure 23: Visual representation of security template process



The process involves the following stages:

1. **Onboarding certificates and keys:** Trustpoints are configured using various onboarding methods (e.g., SCEP, file import) to manage CA certificates, identity certificates, and keys. Certz profiles may also be set up for advanced lifecycle management.
2. **Defining security templates:** An administrator creates a named security template using CLI, specifying settings such as TLS versions, cipher suites, certificate authentication policy, trust anchors, identity sources, and compliance modes.
3. **Assigning templates to applications:** Applications are configured to reference a specific security template rather than embedding their own security settings.
4. **Application registration and initialization:** Each application registers with the security template infrastructure, providing a callback for configuration change notifications and initializing its SSL/TLS context using library APIs.
5. **Policy enforcement during operation:** Applications use the security template's parameters for certificate validation, TLS negotiation, and compliance enforcement during secure communications.
6. **Change notification and dynamic updates:** When a security template or associated certificate material changes, the infrastructure notifies all registered applications, which can then re-initialize their sessions or contexts to apply the new configuration.
7. **Ongoing management:** Administrators can update templates, rotate certificates, and audit compliance centrally, with changes automatically propagated to all consuming applications.

Result

This process enables centralized, consistent, and scalable management of security policies across Cisco IOS XR applications, reducing operational overhead, minimizing misconfiguration risk, supporting rapid adaptation to regulatory requirements, and facilitating secure, compliant communications in large-scale deployments.

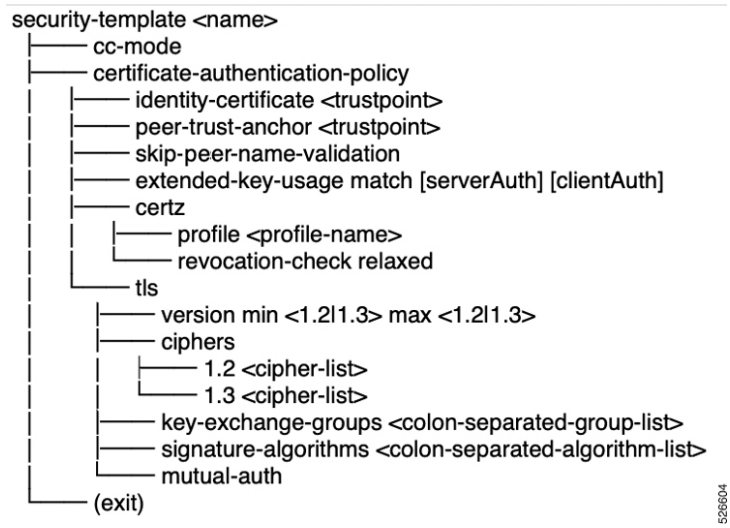
Configure security template

Centralize and standardize the configuration and management of security policies, certificate authentication, and TLS controls for applications running on Cisco routers.

Use a security template to enforce consistent certificate policies, TLS settings, and compliance controls across multiple services and applications, reducing operational overhead and misconfiguration risk.

Configuration concepts and definitions

Figure 24: CLI Structure



Before creating and configuring a security template, you must understand the core concepts and the supported features.

- **Security Template Mode:** You define a template with a unique name.
- **Common Criteria (CC) Mode:** You enable stricter, compliance-focused behavior.
- **Certificate Authentication Policy:** This policy configures how the system handles certificates:
  - **Identity Certificate:** This specifies the trustpoint for the leaf certificate and its chain. The system uses this when a server or client (for mutual TLS or mTLS) must present a certificate chain.
  - **Peer Trust Anchor:** This trustpoint validates server or client certificates. If you configure both a peer identity and a peer trust anchor, the peer trust anchor takes precedence for validating the incoming certificate.
  - **Skip peer name Validation:** This disables hostname verification against the certificate's DNS Subject Alternative Name (SAN) or subject Common Name (CN). Applications typically use this only in lab, testing, or controlled environments.

- **Extended Key Usage Match:** This enforces validation of `serverAuth` and `clientAuth` extensions in certificates. It prevents certificates from misuse outside their authorized purpose.
- **Certz Profile:** This specifies the Certz profile to source the identity certificate, private key, and Certificate Authority (CA) bundle from a gNSI Certz profile. This option takes precedence over trustpoints if configured.
- **Certz Revocation Method Relaxed:** Enabling this option prevents session authentication failure if the Certificate Revocation List (CRL) bundle misses a CRL for any certificate in the verified chain.

- **TLS Settings:** These settings control TLS protocol behavior:
  - **Version Range Control:** You configure the minimum and maximum TLS versions that the system can negotiate (e.g., minimum TLS 1.2, maximum TLS 1.3). This ensures compliance with security policies and prevents downgrade attacks.
  - **Cipher Suites per TLS Version:** You define the allowed set of ciphers for each TLS version (e.g., AES-GCM). This provides fine-grained control to disable weak ciphers.
  - **Key Exchange Groups for Negotiation:** You define the elliptic curves or finite field groups (e.g., x25519, secp256r1, ffdhe3072) used during the TLS key exchange process. This ensures the use of modern and secure Diffie–Hellman parameters.
  - **Signature Algorithms Allow-list:** You specify the permitted digital signature algorithms (e.g., rsa_pss_rsae_sha256, ecdsa_secp256r1_sha256) for TLS handshakes. This prevents the use of deprecated options.
  - **Mutual-auth:** This enables mutual TLS (mTLS).

Before you begin

- Ensure required libraries (`libcrypto_xr`, `libssl_xr`) and header files are available.
- Have necessary trustpoints and/or Certz profiles configured for certificate management.
- Confirm the application supports integration with security templates.

Follow these steps to configure a security template for TLS-enabled applications:

Procedure

Step 1 Enter security template mode by defining a unique template name.

Example:

```
Router(config)# security-template template1
```

Step 2 (Optional) Enable Common Criteria (CC) mode if enhanced compliance is required.

Example:

```
Router(config-security-template)# cc-mode
```

Step 3 Enter certificate authentication policy mode to configure certificate-based settings.

- Specify the identity certificate trustpoint for leaf certificate and chain.
- Specify the peer trust anchor trustpoint for validating incoming certificates.
- (Optional) Skip peer name validation if using in lab or controlled environments.
- (Optional) Enforce extended key usage matching for serverAuth and/or clientAuth.
- (Optional) Specify a Certz profile to source certificates and keys.

Example:

```
Router(config-security-template)# certificate-authentication-policy
Router(config-certificate-authentication-polic)# peer-trust-anchor trustpoint1
Router(config-certificate-authentication-polic)# identity-certificate CA2
Router(config-certificate-authentication-polic)# extended-key-usage match serverAuth
Router(config-certificate-authentication-polic)# skip-peer-name-validation
Router(config-certificate-authentication-polic)# certz-profile certz1
```

Step 4 Enter TLS configuration mode to set protocol and cryptographic parameters.

- Define the minimum and maximum TLS versions allowed (e.g., min 1.2, max 1.3)
- Specify allowed cipher suites for each TLS version as needed.
- Set permitted key exchange groups (elliptic curves or Diffie-Hellman groups).
- Specify allowed signature algorithms for TLS handshakes.
- (Optional) Enable mutual authentication (mTLS) if required.

Example:

```
Router(config-certificate-authentication-polic)# tls
Router(config-tls)# ciphers 1.2 AES256-GCM-SHA384:ECDHE-RSA-AES128-GCM-SHA25 1.3
TLS_AES_256_GCM_SHA384:TLS_AES_128_GCM_SHA256
Router(config-tls)# version min 1.2 max 1.3
Router(config-tls)# key-exchange-groups P-256:P-384:X25519
Router(config-tls)# signature-algorithms
rsa_pss_rsae_sha256:rsa_pss_rsae_sha384:rsa_pss_rsae_sha512:rsa_pkcs1_sha256:rsa_pkcs1_sha384:rsa_pkcs1_sha512
```

Example:

Configuration Example:

```
Router(config)# security-template template1
Router(config-security-template)# cc-mode
Router(config-security-template)# certificate-authentication-policy
Router(config-certificate-authentication-polic)# tls
Router(config-tls)# ciphers 1.2 ECDHE-RSA-AES128-GCM-SHA256 1.3 TLS_AES_256_GCM_SHA384
Router(config-tls)# version min 1.2 max 1.3
Router(config-tls)# key-exchange-groups P-256:P-384:X25519
Router(config-tls)# signature-algorithms rsa_pss_rsae_sha256:ecdsa_secp256r1_sha256
Router(config-tls)# !
Router(config-certificate-authentication-polic)# peer-trust-anchor trustpoint1
Router(config-certificate-authentication-polic)# identity-certificate trustpoint2
Router(config-certificate-authentication-polic)#  extended-key-usage match serverAuth
Router(config-certificate-authentication-polic)# skip-peer-name-validation
Router(config-certificate-authentication-polic)# !
```

Step 5 View running configuration.

Example:

```
Router(config-security-template)# sh run security-template template1

security-template template1
 cc-mode
 certificate-authentication-policy
   tls
    ciphers 1.2 ECDHE-RSA-AES128-GCM-SHA256 1.3 TLS_AES_256_GCM_SHA384
    version min 1.2 max 1.3
    key-exchange-groups P-256:P-384:X25519
    signature-algorithms rsa_pss_rsae_sha256:ecdsa_secp256r1_sha256
  !
  peer-trust-anchor trustpoint1
  identity-certificate trustpoint2
  extended-key-usage match serverAuth clientAuth
  skip-peer-name-validation
 !
!
```

Step 6 Apply the configured security template to desired applications (e.g., syslog, HTTP client) by referencing the template name in their configuration.

Example:

```
Router(config)# logging tls-server TEST
Router(config-logging-tls-peer)# severity debugging
Router(config-logging-tls-peer)# trustpoint tp
Router(config-logging-tls-peer)# address ipv4 10.105.230.83
Router(config-logging-tls-peer)# security-template template1
```

Note When both a standalone trustpoint and a security template are configured under syslog server, the trustpoint specified within the security template overrides the standalone trustpoint.

Step 7 View running configuration.

Example:

```
Router(config-security-template)# sh run logging tls-server TEST

logging tls-server TEST
 severity debugging
 address ipv4 10.105.230.83
 security-template template1
!
```

Step 8 Register the application with the security template infrastructure to receive change notifications and apply updates dynamically.

The security template enforces unified security policies and TLS parameters across all referenced applications. Applications receive and respond to configuration changes automatically, maintaining compliance and reducing risk.

What to do next

- Monitor application logs and debug output for correct application of the security template.
- For certificate rotation or policy updates, update the security template as needed; registered applications will receive notifications and can adapt dynamically.

TLS RFC 5289 compliance for security template

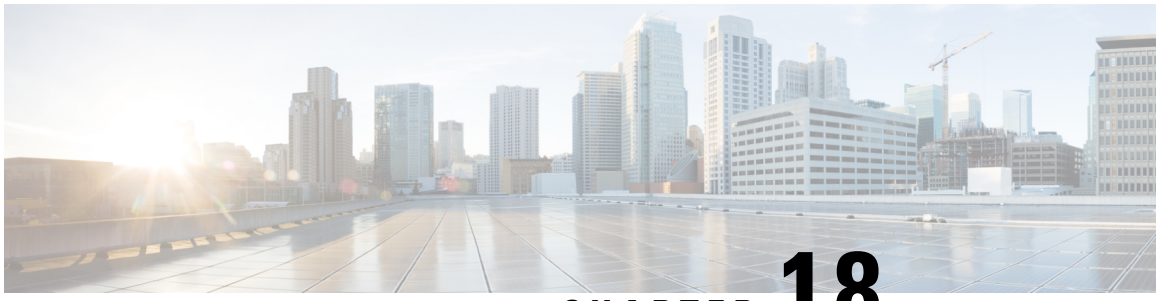
TLS RFC 5289 compliance is a security feature that:

- supports Common Criteria (CC) mode,
- specifies new cipher suites, and
- provides stronger Elliptic Curve Cryptography (ECC) algorithms.

Table 62: Feature History Table

Feature Name	Release Information	Feature Description
TLS RFC 5289 compliance for security template framework	Release 25.4.1	The security template framework is based on RFC 5289, which specifies new cipher suites for the Transport Layer Security (TLS) protocol. This feature supports Common Criteria (CC) mode which is an enhanced security mode that enforces stricter compliance-focused behavior. It enhances TLS security by introducing stronger Elliptic Curve Cryptography (ECC) algorithms.

The security template framework for TLS enabled applications uses RFC 5289 compliance. The RFC moves away from HMAC-SHA1, utilizing SHA-256 and SHA-384 for message authentication codes. For more information on security template framework, see Security template framework for TLS applications.



CHAPTER 18

Gated shell access

A gated shell access is a security feature that

- controls direct, interactive root shell access for supported shell commands,
- requires authorization using a Cisco or customer Consent Token (CT) path before restricting access, and
- stores in HWTAM Secure Object storage.

Consent Token authorization applies to the shell-access restriction operation. This feature uses a separate Consent Token challenge-response workflow for each shell session.

Table 63: Feature History Table

Feature Name	Release Information	Feature Description
Gated shell access	Release 26.3.1	Shell access restrictions require Consent Token authorization for direct root shell access initiated through supported shell commands.

- Key features of gated shell access, on page 479
- How gated shell access works, on page 480
- Restrict shell access using a Cisco-signed Consent Token, on page 481
- Unrestrict shell access using a Cisco-signed Consent Token, on page 482
- Restrict shell access using an owner Consent Token, on page 483
- Unrestrict shell access using an owner Consent Token, on page 484
- Terminate a pending Consent Token authorization request, on page 485
- Access a restricted shell, on page 486

Key features of gated shell access

Use this reference to determine whether a platform supports gated shell access and to identify the associated restrictions and CLI commands.

Key features:

- Gates only direct shell access via supported CLI commands (such as **bash**, **run**, and **attach**). Indirect access through scripts or internal processes is not gated in the initial release.

- It enables restriction or unrestriction of gated shell access, ensuring that only authorized personnel can change shell-access policies.
- Depends on the consent token infrastructure to be present and supported on the platform for operation.
- The restricted-shell selection is stored in an HWTAM secure object.

Table 64: Gated shell access CLI commands

Command	Purpose
platform security shell-access restrict challenge {cisco | owner}	Initiates the Consent Token challenge for the selected Cisco or customer authorization path.
platform security shell-access restrict response <response_string>	Submits the Consent Token response to authorize and complete the shell-access restriction operation.
platform security shell-access restrict terminate-auth	Terminates the pending Consent Token authorization request for shell-access restriction.
platform security shell-access unrestrict challenge {cisco | owner}	Starts the Consent Token authorization flow to remove the restriction.
platform security shell-access unrestrict response <response_string>	Submits the Consent Token response for unrestriction.
platform security shell-access unrestrict terminate-auth	Terminates the pending unrestriction authorization request operation.
show platform security shell-access status	Displays the restricted-shell configuration and handshake status.

The HWTAM Secure Object holds the restricted-shell selection and handshake state. The restricted-shell selection is preserved across reboots and reloads.

How gated shell access works

The process occurs when a user restricts direct interactive shell access on a platform that supports the Consent Token infrastructure.

Summary

When a user restricts shell access, the system records the restriction state in the HWTAM Secure Object. Subsequent supported direct, interactive root shell access attempts check this state and are denied while the restriction is active.

Workflow

These stages describe the gated shell access workflow:

1. A user restricts shell access or requests access to a direct, interactive root shell.
2. The system stores the restricted-shell access state in the HWTAM Secure Object and applies the restriction to supported direct, interactive root shell access.

3. The user must complete the consent token response challenge handshake.
4. The user enters the Consent Token response. When the response succeeds, the system permits one root shell instance for that user session.
5. A user starts the unrestriction authorization flow, submits the response, and the system returns to the default non-gated root shell behavior after successful authorization.

Restrict shell access using a Cisco-signed Consent Token

Use this task to restrict supported direct, interactive root shell access by using a Cisco-signed Consent Token.

Procedure

Step 1 Generate the challenge string on the router.

Example:

```
RP/0/RP0/CPU0:ios# platform security shell-access restrict challenge cisco
Thu Aug 20 06:06:33.888 UTC

+-------------------------------------+
Node location: node0_RP0_CPU0
+-------------------------------------+
Challenge string:
<challenge_string>
```

This produces a challenge string containing the device ID, a nonce, and the requested action.

Step 2 Submit the challenge string to a Cisco TAC engineer. Cisco verifies that the requester is the legitimate device owner and is authorized to perform the requested action. If verified, the TAC engineer provides a signed response string.

Step 3 Paste the response string provided by the TAC engineer when prompted, to install the signed response on the router.

Example:

```
RP/0/RP0/CPU0:ios# platform security shell-access restrict response
Thu Aug 20 06:07:07.090 UTC
************************************************************
Please enter challenge response string for node location node0_RP0_CPU0
************************************************************
<response_string>
Successfully accepted challenge-response for Restrict Shell Access in node0_RP0_CPU0
```

Example:

The router validates the signature and confirms that the device ID and nonce match its own records. If valid, this restricts the direct shell access.

Note

- To access the root shell after restricting shell access, see Access a restricted shell, on page 486.

  If required, to terminate a pending Consent Token authorization request for shell-access restriction, see Terminate a pending Consent Token authorization request, on page 485.

Unrestrict shell access using a Cisco-signed Consent Token

Use this task to unrestrict the supported direct, interactive root shell access by using a Cisco-signed Consent Token.

Before you begin

Ensure that gated shell access is restricted. For more information about restricting shell access by using a Cisco-signed Consent Token, see Restrict shell access using a Cisco-signed Consent Token, on page 481.

Procedure

Step 1 Generate an unrestrict challenge on the router.

Example:

```
RP/0/RP0/CPU0:ios# platform security shell-access unrestrict challenge
Thu Aug 20 06:06:33.888 UTC

+--------------------------------------+
Node location: node0_RP0_CPU0
+--------------------------------------+
Challenge string:
<challenge_string>
```

Step 2 Submit the challenge string to a Cisco TAC engineer. Cisco verifies that the requester is the legitimate device owner and is authorized to perform the requested action. If verified, the TAC engineer provides a signed response string.

Step 3 Paste the response string provided by the TAC engineer when prompted, to install the signed response on the router.

Example:

```
RP/0/RP0/CPU0:ios# platform security shell-access unrestrict response
Thu Aug 20 06:07:07.090 UTC
************************************************************
Please enter challenge response string for node location node0_RP0_CPU0
************************************************************
<response_string>
Successfully accepted challenge-response for Unrestrict Shell Access in node0_RP0_CPU0
```

The router validates the signature and confirms that the device ID and nonce match its own records. If valid, the response removes the restriction on direct shell access.

Note If required, to terminate a pending Consent Token handshake for shell-access restriction, see Terminate a pending Consent Token authorization request, on page 485.

Step 4 Verify the shell-access status.

Example:

```
RP/0/RP0/CPU0:ios# show platform security shell-access status
Thu Aug 20 05:20:01.690 UTC
Restricted shell access enabled: No
```

Restrict shell access using an owner Consent Token

Use this task to restrict supported direct, interactive root shell access by using an owner Consent Token.

Procedure

Step 1 Generate the challenge string on the router.

Example:

```
RP/0/RP0/CPU0:ios# platform security shell-access restrict challenge owner
Thu Aug 20 06:06:33.888 UTC

+-------------------------------------+
Node location: node0_RP0_CPU0
+-------------------------------------+
Challenge string:
<challenge_string>
```

This produces a challenge string containing the device ID, a nonce, and the requested action.

Step 2 To generate the response string owner key type , refer to Provisioning customer consent token guide.

Step 3 Paste the response string provided by the TAC engineer when prompted, to install the signed response on the router.

Example:

```
RP/0/RP0/CPU0:ios# platform security shell-access restrict response
Thu Aug 20 06:07:07.090 UTC
************************************************************
Please enter challenge response string for node location node0_RP0_CPU0
************************************************************
<response_string>
Successfully accepted challenge-response for Restrict Shell Access in node0_RP0_CPU0
```

The router validates the signature and confirms that the device ID and nonce match its own records. If valid, this restricts the direct shell access.

Note

- To access the root shell after restricting shell access, see Access a restricted shell, on page 486.

  If required, to terminate a pending Consent Token authorization request for shell-access restriction, see Terminate a pending Consent Token authorization request, on page 485.

Unrestrict shell access using an owner Consent Token

Restore unrestricted root shell access by removing the restricted-shell setting, allowing access to advanced system features when necessary.

Use this task to unrestrict the supported direct, interactive root shell access by using an owner Consent Token.

Before you begin

Ensure that gated shell access is restricted. For more information about restricting shell access by using an owner Consent Token, see Restrict shell access using an owner Consent Token, on page 483.

Procedure

Step 1 Generate an unrestrict challenge on the router.

Example:

```
RP/0/RP0/CPU0:ios# platform security shell-access unrestrict challenge
Thu Aug 20 06:06:33.888 UTC
+-------------------------------------+
Node location: node0_RP0_CPU0
+-------------------------------------+
Challenge string:
<challenge_string>
```

Step 2 To generate the response string owner key type, refer to Provisioning customer consent token guide.

Step 3 Paste the response string provided by the TAC engineer when prompted, to install the signed response on the router.

Example:

```
RP/0/RP0/CPU0:ios# platform security shell-access unrestrict response
Thu Aug 20 06:07:07.090 UTC
************************************************************
Please enter challenge response string for node location node0_RP0_CPU0
************************************************************
<response_string>
Successfully accepted challenge-response for Unrestrict Shell Access in node0_RP0_CPU0
```

The router validates the signature and confirms that the device ID and nonce match its own records. If valid, the response removes the restriction on direct shell access.

Note If required, to terminate a pending Consent Token handshake for shell-access restriction, see Terminate a pending Consent Token authorization request, on page 485.

Step 4 Verify the shell-access status.

Example:

```
RP/0/RP0/CPU0:ios# show platform security shell-access status
Thu Aug 20 05:20:01.690 UTC
Restricted shell access enabled: No
```

Terminate a pending Consent Token authorization request

Use this task to terminate a pending Consent Token authorization request for restrict or unrestrict shell access.

Use this task only after generating a Consent Token challenge and before submitting a response. The terminate command applies to both Cisco and owner Consent Token requests.

Procedure

Step 1 Generate a Consent Token challenge for either restrict or unrestrict shell access, as described in the applicable task.

Do not submit the response.

Step 2 To terminate a pending Consent Token handshake:

- For restrict shell access, use `platform security shell-access restrict terminate-auth` command.

```
RP/0/RP0/CPU0:ios#
RP/0/RP0/CPU0:ios# platform security shell-access restrict challenge cisco
Thu Aug 20 06:08:45.228 UTC

+--------------------------------------+
Node location: node0_RP0_CPU0
+--------------------------------------+
Challenge string:
<challenge_string>
RP/0/RP0/CPU0:ios# platform security shell-access restrict terminate-auth
Thu Aug 20 06:09:10.334 UTC
Successfully terminated challenge session for Restrict Shell Access in node0_RP0_CPU0
```

- For unrestrict shell access, use `platform security shell-access unrestrict terminate-auth` command.

```
RP/0/RP0/CPU0:ios#
RP/0/RP0/CPU0:ios# platform security shell-access unrestrict challenge cisco
Thu Aug 20 06:08:45.228 UTC

+--------------------------------------+
Node location: node0_RP0_CPU0
+--------------------------------------+
Challenge string:
<challenge_string>
RP/0/RP0/CPU0:ios# platform security shell-access unrestrict terminate-auth
```

```
Thu Aug 20 06:09:10.334 UTC
Successfully terminated challenge session for Unrestrict Shell Access in node0_RP0_CPU0
```

The system confirms that it terminated the pending authorization request.

Access a restricted shell

Use this task to access a restricted, direct, interactive root shell by completing the Consent Token challenge-response workflow.

This task applies to supported direct, interactive root shell commands, such as `bash`, `run` or `attach`.

Procedure

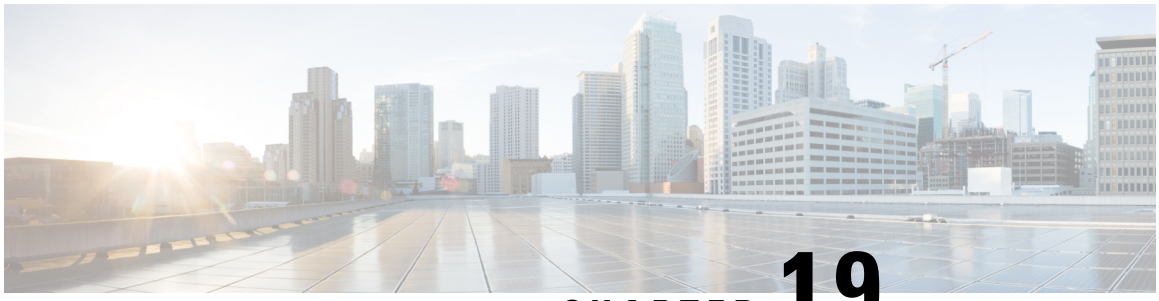
Step 1 Restrict shell access by using either a Cisco-signed Consent Token or an Owner Consent Token.

Step 2 Enter the `bash`, `run` or `attach` command. When prompted, complete the `cisco` or `owner` Consent Token challenge-response.

A valid response permits the requested shell instance.

Example:

```
RP/0/RP0/CPU0:ios# bash
Thu Aug 20 06:01:15.407 UTC
Restricted shell access is enabled. Please obtain a consent token to proceed.
Consent token challenge: <challenge_string>
Enter challenge response, or Ctrl-C to terminate:
<response_string>
Challenge response accepted successfully.
```



CHAPTER 19

SSD Encryption

This module gives an overview of SSD Encryption.

- SSD Encryption, on page 487
- Encrypted Logical Volume, on page 489
- SSD Binding, on page 490
- Data Zeroization, on page 490

SSD Encryption

Table 65: Feature History Table

Feature Name	Release Information	Feature Description
SSD Encryption	Release 7.5.1	This feature enables trust and security in the system's steady state by encrypting data at the disk level. The encrypted data can be accessed *only* with a specific key stored in the TAm.

Customers are concerned about the security of sensitive data present on persistent storage media. User passwords are limited in their capability to protect data against attackers who can bypass the software systems and directly access the storage media.

In this case, only encryption can guarantee data confidentiality.

Cisco IOS XR Software Release 7.5.1 introduces SSD encryption that allows encrypting data at the disk level. SSD encryption also ensures that the encrypted data is specific to a system and is accessible *only* with a specific key to decrypt them.

Data that can be encrypted is sensitive information such as, topology data, configuration data, and so on.

Encryption can be achieved through:

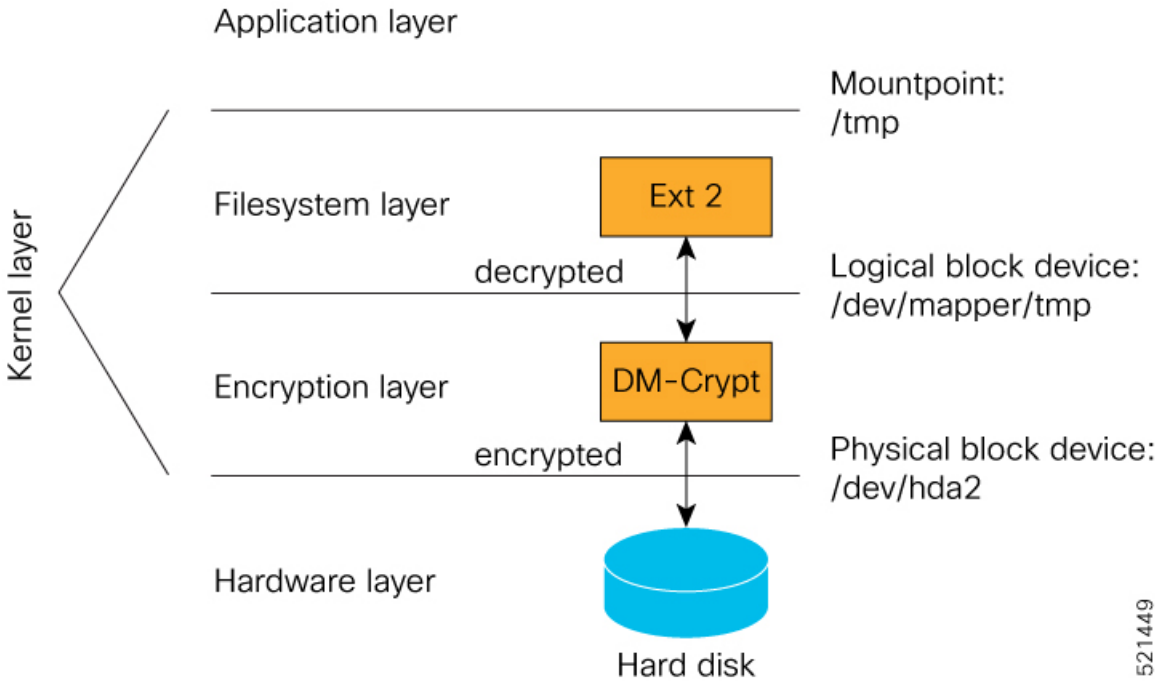
- DM-Crypt
- CPU with AES-NI support
- CryptSetup

DM-Crypt

DM-Crypt is a Linux kernel module that provides disk encryption. The module takes advantage of the Linux kernel's device-mapper (DM) infrastructure. The DM provides a way to create virtual layers of block devices.

DM-crypt is a device-mapper target and provides transparent encryption of block devices using the kernel crypto API. Data written to the block device is encrypted; whereas, data to be read is decrypted. See the following figure.

Figure 25: DM-Crypt Encryption

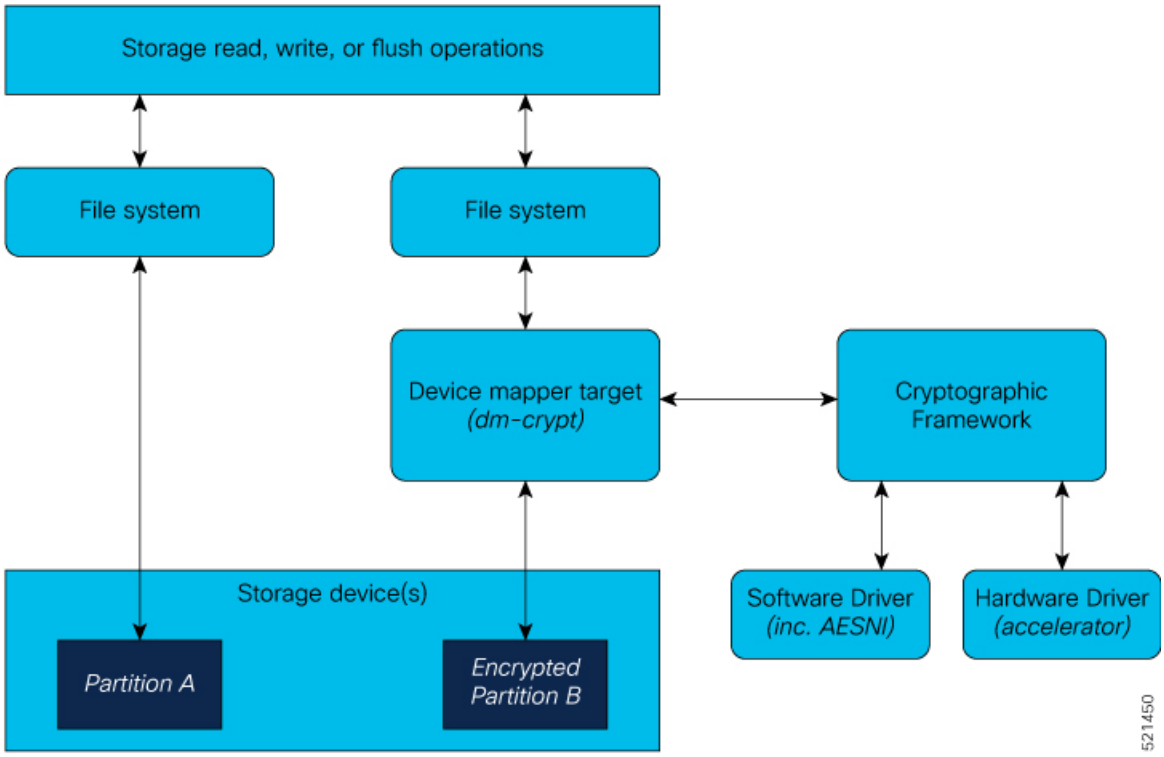


AES-NI Support

Intel's Advanced Encryption Standard New Instructions (AES-NI) is a hardware-assisted engine that enables high-speed hardware encryption and decryption. This process leaves the CPU free to do other tasks.

When the input-output operations are started, the read-write requests that are directed at the encrypted block device are passed to the DM-Crypt. DM-Crypt then sends multiple cryptographic requests to the Cryptographic Framework. The crypto framework is designed to take advantage of off-chip hardware accelerators and provides software implementations when accelerators are not available. See the following image.

Figure 26: AES-NI Support



CryptSetup

DM-Crypt relies on user space tools, such as cryptsetup to set up cryptographic volumes. Cryptsetup is a command-line-interface (CLI) tool that interacts with DM-Crypt for creating, accessing, and managing encrypted devices.

Encrypted Logical Volume

An encrypted logical volume (LV) can be created during software installation.

You can activate or deactivate the encrypted disk partition on demand. In addition to being activated, all sensitive files are also migrated from the unencrypted disk partition to the encrypted disk partition. The encrypted files can be migrated back during deactivation.

You can activate the data encryption by using the `disk encryption activate` *`location`* command.

The encrypted logical volume capacity is 150MB of disk space and is available as /var/xr/enc for applications to access.

Note Although applications can choose to use this space for storage, that data is not be part of data migration if the software image is downgraded to a version that does not support encryption.

SSD Binding

When encryption is activated on a system, each card on the system generates a random encryption key and stores it in its own secure storage—the Trust Anchor module (TAm). During successive reboots, the encryption key is read from the TAm and applied to unlock the encrypted device. Since each card stores its encryption key locally on the TAm, an SSD that is removed from one card and inserted into another cannot be unlocked by the key stored on that card, thereby making the SSD unusable.

If encryption is activated, the encrypted LV can only be unlocked by using the key stored in the TAm. So, if an encrypted SSD is removed and moved to another line card, the SSD cannot be unlocked. In other words, when you activate encryption, the SSD is bound to the card it is inserted in.

Data Zeroization

Zeroization refers to the process of deleting sensitive data from a cryptographic module.

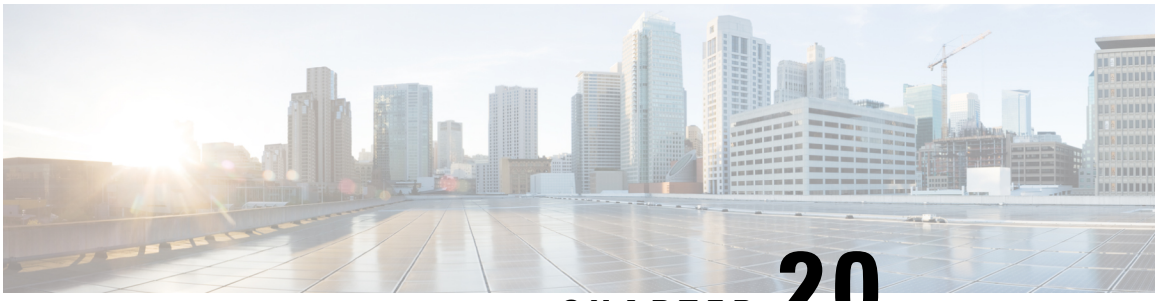


Note In case of a Return Material Authorization (RMA), you must *factory reset* the data.

You can perform zeroization by using the `factory reset` *`location`* command from the XR prompt.



Caution Running this command while encryption is activated, deletes the master encryption key from the TAm and renders the motherboard unusable after the subsequent reload.



CHAPTER 20

Cisco MASA Service

Table 66: Feature History Table

Feature Name	Release Information	Feature Description
Cisco MASA Service	IOS XR 7.8.1	The Cisco Manufacturer Authorized Signing Authority (MASA) service creates ownership vouchers (OVs) for a Cisco IOS XR router. These OVs along with the owner certificate (OC) **certify** that the router belongs to a given customer. Use cases where OVs and OCs are required include secure ZTP workflows and securely booting up your device on a 5G cell site over a third-party ethernet service. You can use the MASA service to download, and view logging and audit of OVs for the routers you own. This service also enables Cisco's Account teams to assign the serial number of a device to customers and view details of the logging, verification, and audit of OVs.

Key Terms and Concepts

Authentication Flow: The purpose of the Authentication flow is to identify and authenticate the router when it boots up. During this flow, the router also checks if the network can be trusted. The router does this by:

- validating the OV it received during the bootstrapping process and
- verifying the signature on the onboarding information with the owner certificate it received during the bootstrapping process.

The workflow involves the router booting to dynamically obtain OV from Manufacturer Authorized signing Authority (MASA).

MASA Service: There are many services that require the ownership of the router to be authenticated, so it can be trusted by the network. MASA is a service run by Cisco to create and log OVs that are then used to validate the ownership of the router.

Owner Certificate: The OC is an X.509 certificate [RFC5280] that is used to identify an *owner*, for example, an organization. The OC can be signed by any certificate authority (CA).

The OC is used by a router to verify the CA signature using the public key that is also in the owner certificate.

The OC structure must contain the owner certificate itself, as well as all intermediate certificates leading to the "pinned-domain-cert" (PDC) certificate specified in the ownership voucher.

Ownership Voucher: The ownership voucher (OV) [RFC8366] is used to securely identify the router's owner, as known to the manufacturer. The ownership voucher is signed by the device's manufacturer.

The OV is used to verify that the owner certificate has a chain of trust leading to the trusted certificate (PDC) included in the ownership voucher.

pinned-domain-cert: The PDC field present in the OV typically pins a domain certificate, such as the certificate of a domain CA.

- Why Do I Need Cisco MASA?, on page 492
- Use Cases for Ownership Vouchers, on page 492
- Authentication Flow, on page 493
- Interacting with the MASA Server, on page 495
- Workflow to Provision a Router Using Ownership Voucher, on page 502

Why Do I Need Cisco MASA?

The Cisco MASA service securely authorizes ownership of a router so that the router can then establish a secure connection to the router owner's (your) network infrastructure.

The establishment of the ownership of the router is achieved through an authentication workflow that on successful completion generates an ownership voucher (OV). The primary purpose of the OV is to securely convey a certificate—the "pinned-domain-cert" (PDC), that the router can then use to authenticate subsequent interactions with the network, for example, secure bootstrapping. Establishing ownership is important to the bootstrapping mechanisms so that the router can authenticate the network that is trying to take control of it.

Use Cases for Ownership Vouchers

- **Secure Zero Touch Provisioning (ZTP) Bootstrapping**

  Secure ZTP requires the ability to securely bootstrap a router over an untrusted network. This requires the ability of MASA to provide an OV to the router. The OV is used to authenticate the router to ensure connectivity of the router to the network.

  For more information on Secure ZTP, see the Secure Zero Touch Provisioning chapter in the *System Setup and Software Installation Guide for NCS 5500 Series Routers*.

Note MASA can help generate OVs for Cisco Routers only.

- **Application Hosting on XR**

  Cisco IOS XR's Application Hosting (App Hosting) capability provides an IOS XR container on the router. This allows an application that augments XR features to be deployed. These applications can fall in one of the following categories:

  - Customer Apps—developed by Cisco's customers and cannot be signed by Cisco.
  - Partner Apps—developed by partners and are signed by Cisco.
  - Cisco App—developed by Cisco and signed by Cisco.

  You can use MASA in conjunction with the Golden ISO Tool (gisobuild.py) to provide the OVs to enable secure workflows for onboarding third party RPMs on router running Cisco IOS XR.

  For more information, see the *Application Hosting Guide for Cisco NCS 5500 Series Routers*.

- **Deploy Router Using BootZ**

  Bootz is a secure zero-touch provisioning solution for data centers that automates the setup of network devices while ensuring robust security. It enables devices to connect and authenticate with the Bootz server, safeguarding the onboarding process against unauthorized access and cyber threats, streamlining remote device configuration without compromising safety.

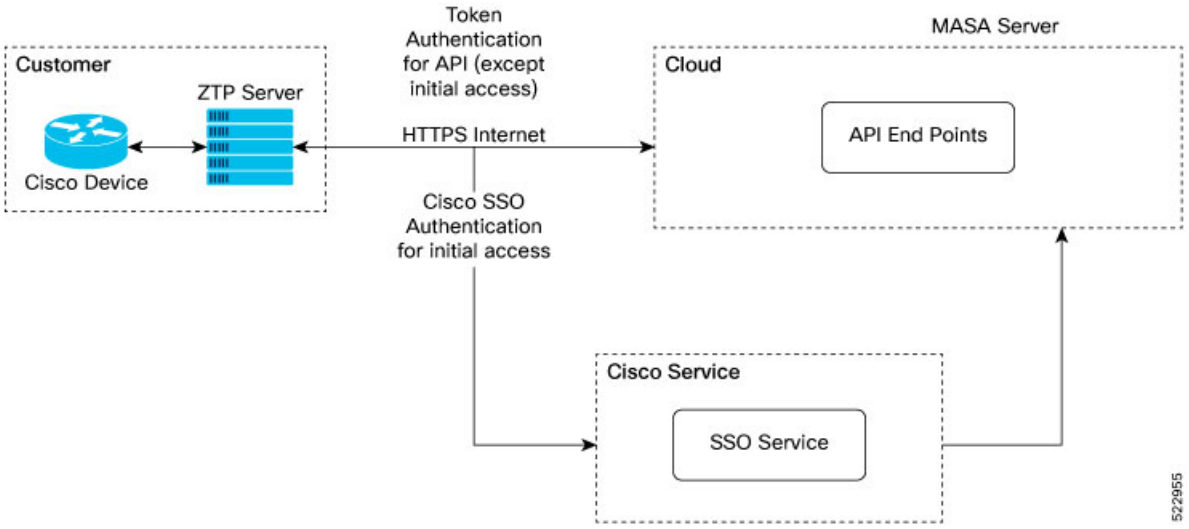
  Bootz uses a MASA to issue OVs that authenticate network devices during zero-touch provisioning.

  For more information on BootZ, see the *System Setup and Software Installation Guide for ASR 9000 Series Routers*.

Authentication Flow

The following figure is a high-level overview of different components involved in the authentication flow.

Figure 27: Components of the Authentication Flow



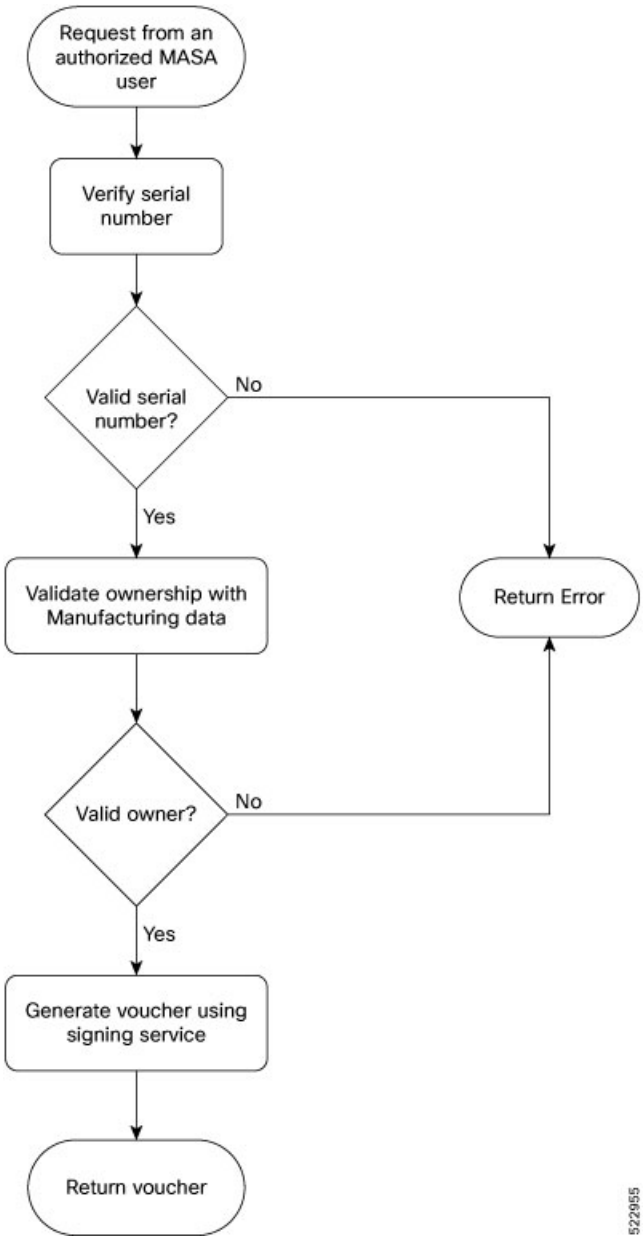
You can interact with the MASA Server web application through the ZTP Server to request, manage, and download the OVs for your routers.

The Zero Touch Provisioning (ZTP) server is to used to make a REST API call to the MASA Server.

The MASA Server authenticates the user, and on successful validation, generates the OVs.

The following figure illustrates the typical workflow to obtain the OVs.

Figure 28: Workflow to Obtain Ownership Vouchers



Interacting with the MASA Server

There are two ways to interact with the MASA server:

- through Web Application
- through REST API calls

Entities

The following entities interact with the MASA Server:

- **Organization**—A group in MASA specific to a Cisco customer. Data and access for each Organization is available to members of that group only.
- **Admin**—One or more initially-designated member(s) of an Organization who can invite other members into that organization in MASA, set access restrictions, and adjust other organization level settings.
- **User**—Any non-admin member of an organization who can interact with MASA. A user must be invited into an organization by the Admin
  - By default, new users have view-only access.
  - The Admin assigns permissions to request, download, or archive ownership vouchers

Prerequisites for Interacting with MASA Server

1. You must be an authorized MASA User
   - You must have a Cisco account and an active invitation to access MASA for the first time.

Note Contact the Cisco Technical Assistance team or your Account team to get a Cisco account.

   - Initial authentication requires *Cisco Single Sign On* to the MASA web application (masa.cisco.com).

     For subsequent authentication, you can generate access keys called *tokens*. Token serve as an alternative authentication mechanism that can be passed along in the header of API calls.

Note To generate access keys for the first time, on masa.cisco.com, go to **Settings**→ **Tokens**. For subsequent sessions, use API calls to manage existing tokens or create new ones as long as an unexpired token is still available.

     The following is an example of using a token in a header of a REST API call.

```
‘Authorization: Bearer
637c98ddcc58c75f679a94d7f244777be05c6600923c4549bc5669b26e04f2bc
gAAAAABjfRr9hqndFqbuqes9OvcfgucApgxpmm9qoVmUidYEs-_Aziu7yue-10dazZ3Rxk6vJHYD2Je7Z-IOD1Zc7kYSuBTX0
6GcQvF2e3nSM-_F9BoltjxAHcXkoMgbqS4APFGi16LiWRyP2b1_0rZO-EaTKFLEldTLfMAmovPDkZZ5vbBwRS058PZN1vB3IZIZ
jftYYYi9H_grazfwnAImjKbQC6tjQw==
```

     Tokens can have a custom validity period of up to six months that can be revoked at any time. The scope of the tokens is limited to scope of your role.

2. ZTP server must be able to access the Internet

Note MASA application is served through HTTPS to provide a secure connection between the end user and the service.

User Permissions

The MASA Server supports Role Based Access Control and provides the following access:

- Regular user—By default, regular users have only read access to their organization. Admin users can provide additional privileges as required.
- Admin—Admin users have the ability to view and manage OVs for all routers in the database in their organization as well as other privileges as mentioned in the table below.

Table 67: User Permissions

Type	Regular User	Admin
Invite other People into the organization	Not allowed	Allowed by default
Add or remove permissions for other users	Not allowed	Allowed by default
View all existing vouchers	Allowed by default	Allowed by default
Request new vouchers	Permission can be provided by Admin	Allowed by default
Download vouchers	Permission can be provided by Admin	Allowed by default
Archive vouchers	Permission can be provided by Admin	Allowed by default

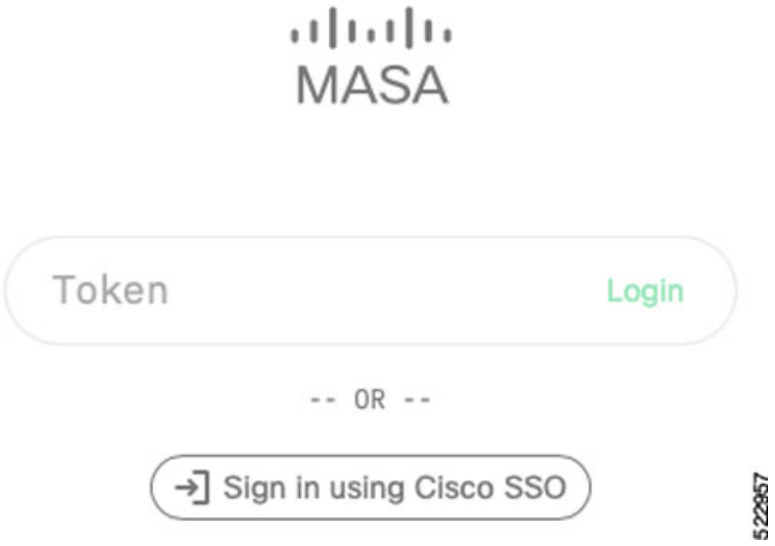
Interacting with MASA Through Web Application

Table 68: Feature History Table

Feature Name	Release Information	Feature Description
Pre-upload Pinned-Domain Certificate	Release 24.1.1	You can now pre-upload your Pinned-Domain Certificate (PDC) credentials before requesting OVs Ownership Vouchers (OVs) from the MASA server, thus making the voucher request process easier.

1. Go to masa.cisco.com

Figure 29: Sign in Page—MASA Web Application



2. Click **Sign in using Cisco** SSO.
3. Enter your username and password to access the application
4. Accept the End User License Agreement.

   The MASA Home page displays the status of any recent requests that were initiated and quick links to download any recently generated ownership vouchers.

Figure 30: Home Page—MASA Web Application



Requesting OVs for Your Router

1. Click **New Request** on the top right of the Home page.
2. In the New Request dialog box, enter details for one of the following:
   - Serial number of your router

     You can get the serial number from the bottom of your router; it is an 11 digit alphanumeric string. You can also get the serial number by running the **show version** command on your router.

- Pinned-domain Certificate

  There are multiple ways to generate a PDC (.pem). For example, through OpenSSL. You can either paste the content of the certificate directly or browse to a file that contains the PDC.

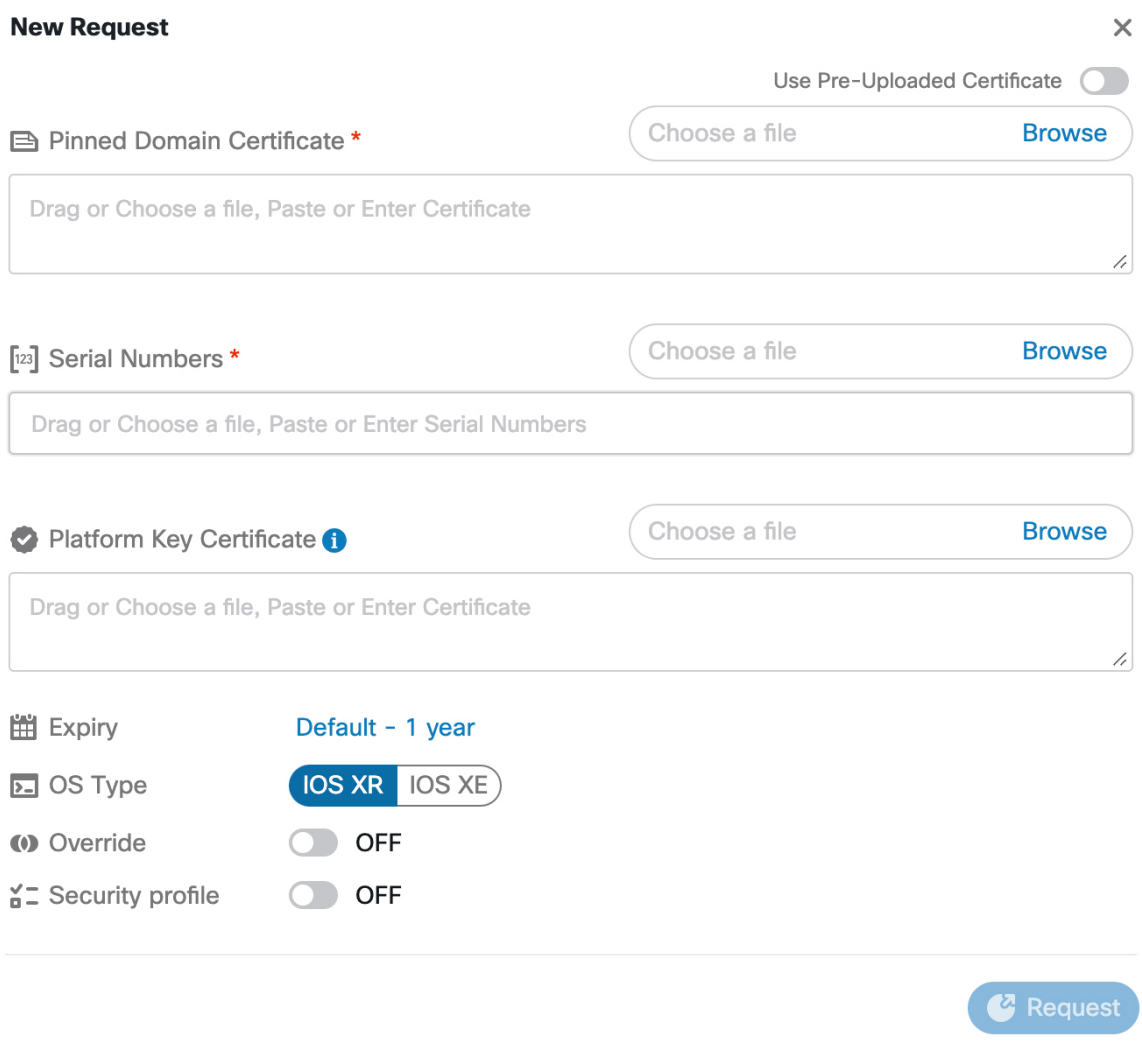
  You can pre-upload the certificate prior to requesting the OV.

  To select the pre-uploaded certificate while requesting OV, turn on the toggle button named *use pre-uploaded certificate*.You can see the already uploaded certificates here, you can select the certificate from this list.

- Serial number of one or more routers for which you want the OVs.

Note Always use the serial number of the route processor (RP) of your router.

Figure 31: New Request Page



523805

Figure 32: Home Page—With New OVs Displayed



522960

Depending on your user permissions, you can perform the following actions from the Home page.

- Download the generated OVs.
- Regenerate OVs.
- View details of past requests
- Filter, sort, and group the requests based on their attributes
- Archive the OVs.

Interacting with MASA Through REST APIs

You can also use APIs to programmatically interact with the MASA service.

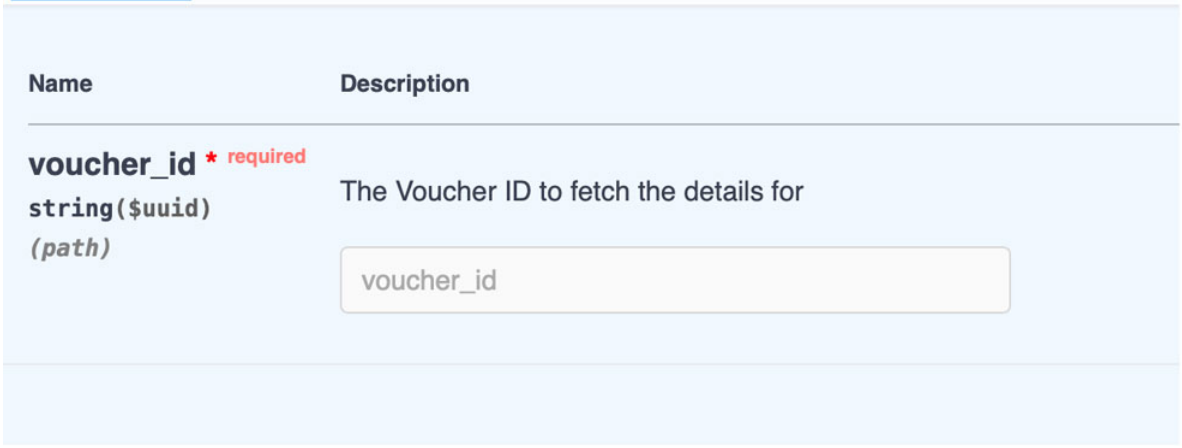
See the OpenAPI documentation page that contains details about the paths, formats, and structures of the APIs.

For example, use this API to request for the ownership voucher:

```
POST /request/ov
```

Use this API to fetch details about an already generated voucher:

```
GET /voucher/{voucher_id}
```




```
Response:
{
  "ok": true,
  "voucher": {
    "req_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
    "voucher_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
    "requested_at": "2022-08-31T09:43:39.719Z",
    "created_at": "2022-08-31T09:43:39.719Z",
    "expires_at": "2022-08-31T09:43:39.719Z",
    "last_renewal_at": "2022-08-31T09:43:39.719Z",
    "assertion": "logged",
    "status": "completed",
    "serial_number": "T8I52J1IKOM",
    "pdc_organization": "Cisco Systems",
    "requested_by": "user1@cisco.com"
  }
}
```

Note “serial Number” is serial number of the route processor. You can provide up to 20 serial numbers in a single request.

Interaction with MASA through gRPC

Table 69: Feature History Table

Feature Name	Release Information	Feature Description
Interaction with MASA through gRPC	Release 24.1.1	From this release, you can use the gRPC protocol to interact with MASA APIs in addition to the current HTTP protocols. Through structured serialization of data with gRPC's Protocol Buffers, the communication between services is made more efficient, type-safe, and consistent.

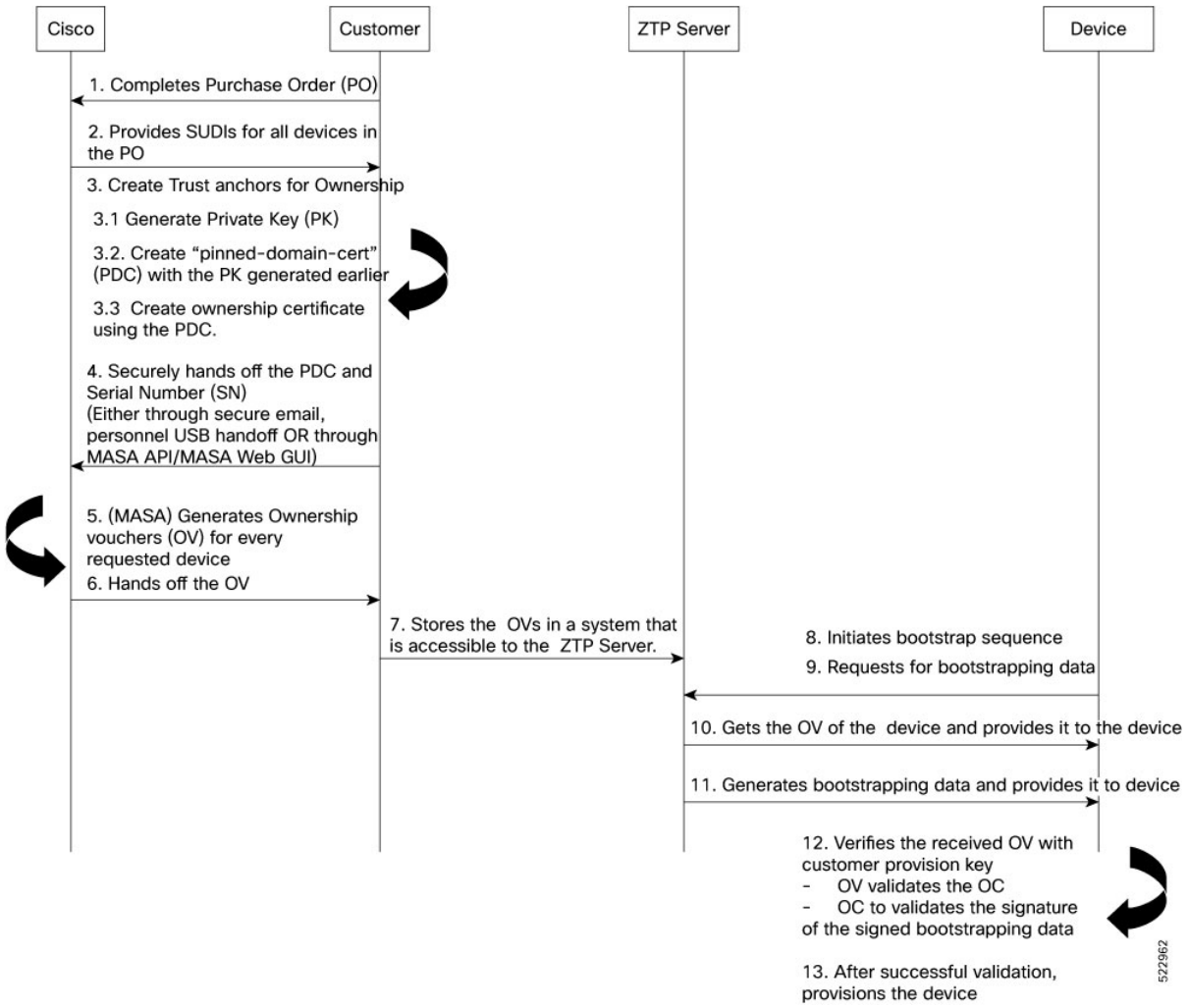
The following MASA APIs are accessible using gRPC protocol in addition to http protocol:

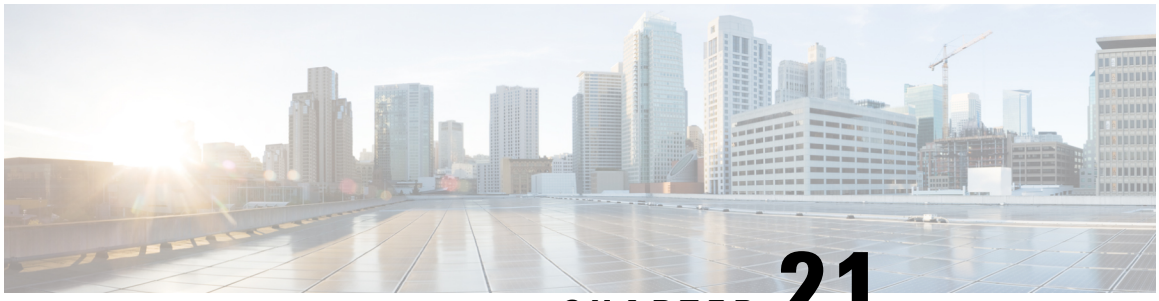
RPC	Description
rpc GetGroup	Returns the domain-certificates (keyed by id), serials, and user/role mappings for that group.
rpc AddUserRole	Assigns a role to a user in a named group. Username is unique to an Org ID.
rpc RemoveUserRole	Removes a role from a user in a named group. Username is unique to an Org ID.
rpc GetUserRole	Returns the roles that the user is assigned in the group. Username is unique to an Org ID. A user can only view roles of another user in the group that it has a role assigned to.
rpc CreateDomainCert	Creates the certificate in the group.
rpc GetDomainCert	Reveals the details of the certificate.
rpc DeleteDomainCert	Deletes the certificate from the database.
rpc GetOwnershipVoucher	Issues an ownership voucher.

For more information on gRPC, see Use gRPC Protocol to Define Network Operations with Data Models in the *Programmability Configuration Guide for ASR 9000 Series Routers.*

Workflow to Provision a Router Using Ownership Voucher

The following figure illustrates the complete workflow to provision a Cisco IOS XR router by using the ownership vouchers.


Cisco
Customer
ZTP Server
Device
1. Completes Purchase Order (PO)
2. Provides SUDIs for all devices in the PO
3. Create Trust anchors for Ownership
3.1 Generate Private Key (PK)
3.2. Create "pinned-domain-cert" (PDC) with the PK generated earlier
3.3 Create ownership certificate using the PDC.
4. Securely hands off the PDC and Serial Number (SN) (Either through secure email, personnel USB handoff OR through MASA API/MASA Web GUI)
5. (MASA) Generates Ownership vouchers (OV) for every requested device
6. Hands off the OV
7. Stores the OVs in a system that is accessible to the ZTP Server.
8. Initiates bootstrap sequence
9. Requests for bootstrapping data
10. Gets the OV of the device and provides it to the device
11. Generates bootstrapping data and provides it to device
12. Verifies the received OV with customer provision key
- OV validates the OC
- OC to validates the signature of the signed bootstrapping data
13. After successful validation, provisions the device
522962



CHAPTER 21

EST protocol for Automated Certificate Provisioning

This feature allows you to enable the EST (Enrollment Over Secure Transport) protocol for all trustpoints while using TLS to secure transport.

- EST protocol for automated certificate provisioning, on page 505
- Configure the EST protocol, on page 510

EST protocol for automated certificate provisioning

Enrollment over Secure Transport (EST) is a digital certificate provisioning protocol that

- enhances security by using TLS for secure communication, and
- automates the renewal of certificates to minimize manual intervention.

Table 70: Feature History Table

Feature Name	Release Information	Feature Description
EST protocol for automated certificate provisioning	Release 25.1.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q200, P100], 8700 [ASIC: P100, K100]); Centralized Systems (8600 [ASIC:Q200]) ; Modular Systems (8800 [LC ASIC: Q100, Q200, P100]) This release introduces support for Enrollment over Secure Transport (EST), a digital certificate provisioning protocol, which enhances certificate management by offering secure transport using TLS and designated certificate requestors. It enables automated certificate renewal. EST is an enhancement of the existing Simple Certificate Enrollment Protocol (SCEP), providing improved security and flexibility for certificate management operations over both IPv4 and IPv6 networks. The feature introduces these changes: **CLI**: • The **method-est** keyword is introduced in the **crypto ca trustpoint** command. • The **client-authentication** command is introduced. • The **SSL-profile** command is introduced. **YANG Data Model**: • `Cisco-IOS-XR-crypto-cepki-cfg` • `Cisco-IOS-XR-um-crypto-cfg` (see GitHub, YANG Data Models Navigator)

EST protocol—key features and benefits

These are the key features and benefits of the EST protocol:

- **Secure transport**—EST utilizes TLS to ensure that all messages and certificates are securely transmitted without the need for additional encapsulation.
- **Designated certificate requestors**—With EST, the certificate signing request (CSR) can be associated with a specific trusted requestor that is authenticated with TLS.
- **Automated certificate renewal**—The protocol supports automatic re-enrollment, facilitating seamless renewal of certificates.

EST protocol—key components and related terminologies

Understand key components and terms related to the EST protocol:

- **Certificate Management Protocol (CMP)**—A protocol with comprehensive management capabilities for managing digital certificates, including their enrollment, renewal, and revocation.
- **Certificate signing request (CSR)**—A message sent by the client to the CA to request a digital certificate. It includes the client's public key and identifying information.
- **PKCS 7**—A standard used for cryptographic message syntax, which is employed by SCEP to encapsulate messages.
- **Public Key Infrastructure (PKI)**—The framework that manages digital certificates and keys, essential for certificate provisioning operations using the EST protocol.
- **Simple Certificate Enrollment Protocol (SCEP)**—An older protocol used for certificate provisioning that relies on HTTP and PKCS 7 for securing messages and lacks some of the modern security features of EST.

Client authentication options in the EST protocol

- TLS certificate-based authentication (mTLS)—Requires the client to have a valid certificate chain signed by a CA in the EST server trust store so that server can verify the client during mutual TLS (mTLS) authentication. The bootstrap certificate is used only during the first enrollment and can be onboarded using existing methods like SCEP. For re-enrollment, the certificate issued during the initial enrollment is used in the mTLS handshake.
- HTTP-based authentication—When a bootstrap certificate is unavailable for initial enrollment, the EST client establishes a TLS session and then sends HTTP authentication credentials. This HTTP-based authentication method uses Type6 encryption to protect passwords configured on the router, preventing unauthorized access. By default, Type6 encryption is disabled and must be enabled manually. Additionally, a master key must be created and securely stored in the Trust Anchor Module (TAM). During this process, the client transmits a username and clear text password through the TLS-encrypted channel, ensuring the password remains secure due to the TLS protection. To validate the server during this TLS session, the device must have the server certificate's CA/subordinate CA chain installed locally. This validation confirms the authenticity of the EST server and secures the communication channel before the HTTP authentication details are sent..

Guidelines and limitations for configuring the EST protocol

Supported key types

- In IOS-XR release 25.1.1, only RSA keys are supported for signing the Certificate Signing Request (CSR) during enrollment; ECDSA keys are not.

Client authentication methods

- The EST client supports both TLS certificate-based authentication and HTTP-based authentication. Cisco recommends using TLS certificate-based authentication as per RFC 7030, even though both methods are available.

TLS validation and authentication requirements

- If client or server certificate validation fails during the TLS handshake, EST enrollment fails. No fallback mechanism switches to HTTP authentication if TLS certificate-based client authentication fails.

Re-enrollment

- For re-enrollment, the client always uses the previously issued certificate to establish the TLS connection.
- A re-enrollment profile is required in scenarios where the EST server uses a fixed username and password for re-enrollment, but uses a one-time password (OTP) for initial enrollment. This behavior is optional and depends on how the EST server is configured.

EST Client Support Requires Multiple Trustpoints

The EST client support requires multiple trustpoints because it enables secure certificate enrollment over TLS for all trustpoints configured on the device, and a single trustpoint configuration is insufficient to support this functionality.

Certificate types and their uses in EST protocol

Learn about the types of certificates used in the EST protocol, along with their issuers and specific uses.

Table 71: Certificates and their corresponding uses

If the certificate type is...	And the issuer is...	Then it is used for...
EST server certificate	EST server	authenticating the CA during the TLS handshake when the EST server interacts with clients.
EST client certificate	EST client	authenticating to the EST server during certificate enrollment operations.
End entity or leaf certificate	EST server	non-EST uses, such as authenticating devices in different contexts beyond EST operations.
EST server certificate	third-party web CA	providing authentication for the EST server by a CA that is not part of the EST infrastructure but trusted as a third party.

If the certificate type is...	And the issuer is...	Then it is used for...
third-party EST client certificate	third-party web CA	authenticating the EST client to the EST server for initial interactions before enrollment.

Trust anchor databases in EST protocol

Effective certificate validation and authentication in the EST protocol involve different types of trust anchor (TA) databases, each serving specific purposes.

A TA database refers to a repository of trusted anchor certificates used to verify the authenticity of a Certification Authority (CA) during the certificate enrollment process. The database acts as a trusted source for validating the CA issuing certificates to clients through the EST protocol.

Table 72: Trust anchor databases and their uses

If the TA database type is...	Then it is used by...	For the purpose of...
EST server	EST server	authenticating certificates issued by the EST CA during enroll and re-enroll operations.
EST server (implicit)	EST server	authenticating certificates issued by third-party TAs; can be disabled if not needed.
EST client	EST client	authenticating EST server certificates issued by the EST CA.
EST client (implicit)	EST client	authenticating an EST server using an externally issued certificate; can be disabled if unnecessary.

Pre-defined message types in EST protocol

The EST protocol includes several pre-defined message types, each serving distinct functions to facilitate secure certificate provisioning:

- **CACerts**—This message type is used to request the CA certificates. It ensures the client can validate the CA's authenticity before proceeding with certificate enrollment.
- **SimpleEnroll**—Utilized for simple enrollment of a certificate, this message type involves a client sending a certificate signing request (CSR) to the CA. If the request is valid, the CA issues a certificate in response.
- **Reenroll**—Designed for requesting certificate renewal or reissuance, this message type allows clients to obtain a new certificate with updated validity or information when an existing certificate is about to expire.

Configure the EST protocol

Use this procedure to configure the EST protocol on your router, enabling secure and automated certificate provisioning.

Before you begin

Before configuring the EST protocol, ensure

- access to the EST server is established and operational,
- required certificates and keys are available for configuration,
- you meet specific encryption requirements if you must use HTTP-based authentication:
  - You must create a rsa key using the **crypto key generate rsa key** *key-name* command. Use the generated key to configure the EST_TRUSTPOINT.
  - Type6 encryption is enabled using the command **password6 encryption aes** for secure password handling, and
  - You must create a master key and store it in Trust Anchor Module (TAM) using the command **key config-key password-encryption** to facilitate encryption.

Note A master key for Type 6 encryption is typically stored securely within the device internal memory, specifically within a protected area known as the TAM.

Procedure

Step 1 Select the authentication method.

You can configure either TLS certificate-based authentication, HTTP-based authentication, or both. Use the table below to determine the appropriate sub-step based on the availability of a bootstrap certificate.

Table 73: Bootstrap certificate availability and authentication method

If...	Then follow...
A bootstrap certificate is not available	HTTP-Based Authentication sub-step.
A bootstrap certificate is available	TLS-Based Authentication sub-step.

a) For HTTP-based authentication, create client authentication profiles for enrollment and re-enrollment.

Example:

```
Router(config)#client-authentication profile P1_enroll
Router(config-client-authentication-profile)#username estuser password Encrypt6 estpwd_enrol
Router(config-client-authentication-profile)#exit
Router(config)#client-authentication profile P2_re-enroll
```

```
Router(config-client-authentication-profile)#username estuser password Encrypt6 estpwd_re_enrol
Router(config-client-authentication-profile)#commit
Router(config-client-authentication-profile)#exit
Router(config)#crypto ca trustpoint EST_TRUSTPOINT
Router(config-trustp)#method est
Router(config-trustpoint)#enrollment url https://192.168.30.1:port
Router(config-trustpoint)#authentication-profile P1_enroll
Router(config-trustpoint)#re-enrollment authentication-profile P2_re-enroll
Router(config-trustpoint)#rsakeypair est-client
Router(config-trustpoint)#commit
```

You can use ipv4, ipv6, and hostname addresses as enrollment URL.

Table 74: Re-enrollment profile scenarios for HTTP-based authentication

If...	Then...
separate profiles are configured for enrollment and re-enrollment	*P1_enroll* is used for enrollment and *P2_re-enroll* is used for re-enrollment.
only the enrollment profile is configured	*P1_enroll* is used for both enrollment and re-enrollment.

If you are using the multi-tier CA certificate system, the CA root certificate need to be imported in another trustpoint using terminal, SCEP or local file method.

b) For TLS-based authentication, configure an SSL profile with the bootstrap certificate.

Example:

```
Router# configure terminal
Router(config)# ssl profile EST_SSL_profile
Router(config-SSL-profile)# certificate EST_BOOTSTRAP_TP
Router(config-SSL-profile)# commit
```

For initial enrollment, ensure you add a bootstrap certificate using existing methods like terminal, SCEP, and so on.

Step 2 Configure a PKI trustpoint to specify the enrollment method as EST and attach the authentication profiles.

The code example includes CLI commands to attach both HTTP and TLS-based authentication profiles.

This command initializes the EST method for the specified trustpoint.

Example:

```
Router#configure terminal
Router(config)#crypto ca trustpoint EST-TP
Router(config)#ssl-profile EST_enroll
Router(config-trustp)#method est
Router(config-method-est)#commit
Router(config-trustp)#enrollment authentication-profile P1_enroll
Router(config-trustp)#re-enrollment authentication-profile P2_re-enroll
Router(config-trustp)#commit
```

Step 3 Authenticate and enroll the certificates.

Run the CLI commands in EXEC mode.

```
Router# crypto ca authenticate EST_TP
Router# crypto ca enroll EST_TP
```

Step 4 Verify the certificate enrollment using EST is successful.

Example:

```
Router# show run crypto ca trustpoint EST_TP:
crypto ca trustpoint est_tp
 ssl-profile example
 method est
 !
 enrollment authentication-profile enroll
 re-enrollment authentication-profile re_enroll
!
```

Step 5 Verify the certificates are enrolled on the trustpoint.

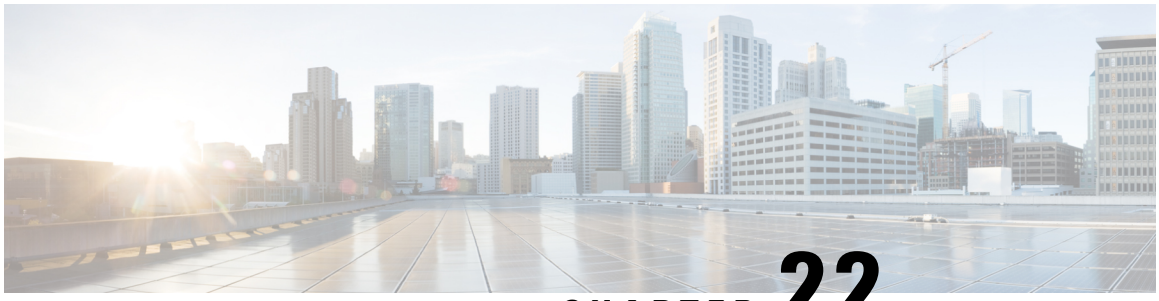
Example:

```
Router# show crypto ca certificates EST_TP

Trustpoint       : EST_TP
================================================
CA certificate
  Serial Number  : 10:01
  Subject:
            CN=SUB_CA_CERT,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Issued By      :
            CN=TWO-LEVEL-CA,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Validity Start : 12:31:40 UTC Sun Jun 14 2020
  Validity End   : 12:31:40 UTC Wed Jun 12 2030

  CRL Distribution Point
            http://10.105.236.78/crl.der
  SHA1 Fingerprint:
             D8E0C11ECED96F67FDBC800DB6A126676A76BD62
Trusted Certificate Chain
  Serial Number  : 0F:A0:06:7A:C9:5E:A9:E7:61:A2:B9:2B:27:D1:D6:8F:3D:51:43:3B
  Subject:
            CN=TWO-LEVEL-CA,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Issued By      :
            CN=TWO-LEVEL-CA,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Validity Start : 13:12:32 UTC Sun Jun 07 2020
  Validity End   : 13:12:32 UTC Sat Jun 02 2040

  CRL Distribution Point
            http://10.105.236.78/crl.der
  SHA1 Fingerprint:
             08E71248FB7578614442E713AC87C461D173952F
Router certificate
  Key usage      : General Purpose
  Status         : Available
  Serial Number  : 28:E5
  Subject:
            CN=test
  Issued By      :
            CN=SUB_CA_CERT,OU=SPBU,O=CSCO,L=BGL,ST=KA,C=IN
  Validity Start : 08:49:54 UTC Mon Feb 06 2023
  Validity End   : 08:49:54 UTC Wed Mar 08 2023
  SHA1 Fingerprint:
             6C8644FA67D9CEBC7C5665C35838265F578835AB
Associated Trustpoint: EST_TP
```



CHAPTER 22

Upgrade CURL libraries for HTTP client

A libcurl upgrade is a feature that

- updates the embedded libcurl library from version 7.30 to 8.17.0 to align with upstream security and protocol improvements,
- preserves specific customizations such as VRF awareness, XR SSL integrations, and TCP performance tuning, and
- ensures backward compatibility and automatic benefit for all dependent applications without user intervention.

Table 75: Feature History Table

Feature Name	Release Information	Feature Description
Upgrade CURL libraries for HTTP client	Release 26.3.1	You gain enhanced security and improved reliability through an upgraded embedded libcurl library. This transparent update preserves existing functionality and continues to support HTTP/1.0 and HTTP/1.1.

- Key benefits for libcurl upgrades, on page 513
- Support and restrictions for libcurl upgrade, on page 514
- How libcurl upgrade works, on page 514

Key benefits for libcurl upgrades

The following are significant attributes and considerations for the libcurl upgrade:

- Updates the libcurl version to 8.17.0 for the underlying HTTP stack.
- Maintains compatibility for all existing APIs, so no changes are required for dependent applications.
- Retains and independently manages platform-specific patches, such as VRF, SSL, and TCP optimizations, by rebasing and validating them against the upgraded libcurl version, simplifying future maintenance.
- Supports major features such as Smart Licensing, Shell utility (Copy), Install Manager, CEPKI, SKS, and Zero Touch Provisioning (ZTP).

- Is not user-configurable and cannot be disabled; no new CLI or configuration is introduced.

Table 76: Libcurl upgrade attribute comparison

Attribute	Previous implementation	Upgraded implementation
Libcurl version	7.30	8.17.0
Supported HTTP protocols	HTTP, HTTPS	HTTP, HTTPS
XR-specific customizations	VRF support, SSL integration (XR libcrypto), TCP tuning	Maintained
API compatibility	Legacy APIs supported	All existing APIs maintained
Configuration impact	None	None (not configurable)



Note The upgrade is completely transparent to users and feature owners. No CLI changes or configuration are required, and the previous libcurl version cannot be restored by configuration.

Support and restrictions for libcurl upgrade

Provides an overview of the Cisco platforms that support the libcurl upgrade, details operational or configuration restrictions, and explains how the enhancement integrates with system infrastructure in supported environments.

The following points summarize platform support and key restrictions for the libcurl upgrade:

- The upgrade applies only to platforms that run Cisco IOS XR software, including supported hardware and virtual platforms that use the upgraded HTTP infrastructure.
- Users cannot enable, configure, or roll back the upgrade. The system image manages the upgrade as a transparent infrastructure enhancement.
- The upgrade preserves VRF awareness and compatibility with XR-specific socket, SSL, and networking features across supported platform variants.
- Existing HTTP client APIs and applications continue to operate without change on supported platforms.

How libcurl upgrade works

The libcurl upgrade is an internal infrastructure improvement that replaces the embedded libcurl library version 7.30 with version 8.17.0. This change aligns the HTTP client stack with modern standards for protocols, security, and maintainability. The upgrade is transparent to users; no changes to commands or configuration are required. Application components continue to function as before, preserving backward compatibility. Security updates, platform-specific features, and operational observability are maintained through integration of upstream fixes and Cisco patches.

- The upgrade is internal and does not alter configuration workflows.

- All existing APIs and application interactions continue to be supported.
- Security and feature enhancements are introduced through targeted patches to the upgraded library.

Summary

The HTTP client infrastructure processes HTTP and HTTPS transactions through coordinated components that use a stable API and asynchronous communication.

- Application component: Initiates HTTP or HTTPS operations by calling the HTTP client library API.
- HTTP client library: Serves as the API interface and manages communication with the HTTP client process.
- HTTP client process: Handles request and response states, session management, concurrency, and calls libcurl.
- libcurl library: Executes protocol operations and network communication, with enhancements for XR integration.

Workflow

These stages describe the transaction flow for HTTP requests by using the upgraded libcurl library:

1. An application component, such as Smart Licensing, Install Manager, Zero Touch Provisioning, or the Copy utility, starts an HTTP or HTTPS request by calling the HTTP client library API.
   - The API remains unchanged, so existing feature components continue to work without modification.
   - The HTTP client library receives and processes the request.
2. The HTTP client library packages the request, adds required parameters, and sends the request asynchronously to the HTTP client process.
   - Asynchronous interprocess communication prevents operations from blocking.
   - Request details, such as VRF information and metadata, are included for execution.
   - The HTTP client process receives the prepared request.
3. The HTTP client process manages the request by queuing transactions and maintaining session state for multiple requests.
   - The process uses the libcurl multi interface to scale concurrent requests efficiently.
   - The process performs error handling and allocates required system resources.
   - Requests are sequenced and validated before dispatch to the communication layer. The process then prepares to call libcurl for protocol execution.
4. The HTTP client process calls the `curl_multi_perform` API for efficient asynchronous network operations.
   - Handles XR-custom socket management, VRF context, and SSL using Cisco extensions.
   - Manages concurrency by using `select` and the libcurl multi interface.

This stage provides reliable, efficient transaction processing. Network protocol negotiation and TLS setup occur in the updated, security-patched libcurl library.

5. The libcurl library establishes the network connection, negotiates SSL or TLS as needed, and completes the HTTP transaction by using HTTP/1.0 or HTTP/1.1.
   - Maintains socket management, SSL via `libssl_xr`/`libcrypto_xr`, and TCP settings.
   - Security patches and protocol improvements are applied in this stage.
   - The library sends response and transaction state information to the HTTP client process.
6. The HTTP client process parses the libcurl response, updates transaction status, logs actions or errors, and tracks operational state.
   - Uses standardized error codes and logging to simplify troubleshooting and monitoring.
   - Makes status information available through show commands.

   Transaction results are passed to the client library. The HTTP client library supplies the response and status to the application.
7. The application receives the response from the HTTP client library and continues normal operation.

   The HTTP transaction process completes, and resources are released. No user or application-level awareness or adaptation is needed for the upgrade.

Result

The system benefits from a secure, up-to-date HTTP client infrastructure with improved protocol support, stronger security, and better operational efficiency, with no change to applications or user experience.