CHAPTER 3

Implementing Keychain Management

Implementing Keychain Management
Restrictions for Implementing Keychain Management
Configure Keychain
Configure Tolerance Specification to Accept Keys
Configure Key Identifier for Keychain
Configure Text for Key String
Determine Valid Keys
Configure Keys to Generate Authentication Digest for Outbound Application Traffic
Configure Cryptographic Algorithm
Lifetime of Key

CHAPTER 4

Implementing URPF

Understanding URPF
Configuring URPF Loose Mode
CHAPTER 5
Implementing Management Plane Protection
- Implementing Management Plane Protection
- Benefits of Management Plane Protection
- Restrictions for Implementing Management Plane Protection
- Configure Device for Management Plane Protection for Inband Interface
- Configure Device for Management Plane Protection for Out-of-band Interface
- Information About Implementing Management Plane Protection
  - Peer-Filtering on Interfaces
  - Control Plane Protection
  - Management Plane

CHAPTER 6
Traffic Protection for Third-Party Applications
- gRPC Protocol
- Limitations for Traffic Protection for Third-Party Applications
- Prerequisites for Traffic Protection for Third-Party Applications Over GRPC
- Configuring Traffic Protection for Third-Party Applications
- Troubleshooting Traffic Protection for Third-Party Applications

CHAPTER 7
Implementing Secure Shell
- Implementing Secure Shell
- Prerequisites to Implementing Secure Shell
- Restrictions for Implementing Secure Shell
- Configure SSH
- Configure SSH Client
- Information About Implementing Secure Shell
  - SSH Server
  - SSH Client
  - SFTP Feature Overview
  - RSA Based Host Authentication
  - RSA Based User Authentication
  - SSHv2 Client Keyboard-Interactive Authentication

CHAPTER 8
Implementing Lawful Intercept
Information About Lawful Intercept Implementation 89
Prerequisites for Implementing Lawful Intercept 90
Restrictions for Implementing Lawful Intercept 91
Lawful Intercept Topology 91
Benefits of Lawful Intercept 92
Installing Lawful Intercept (LI) Package 93
  Installing and Activating the LI Package 93
  Deactivating the LI RPM 93
How to Configure SNMPv3 Access for Lawful Intercept 93
  Disabling SNMP-based Lawful Intercept 94
  Configuring the Inband Management Plane Protection Feature 94
  Enabling the Lawful Intercept SNMP Server Configuration 95
Additional Information on Lawful Intercept 95
  Interception Mode 95
  Data Interception 95
  Scale or Performance Values 96
  Intercepting IPv4 and IPv6 Packets 96
    Lawful Intercept Filters 96
      Encapsulation Type Supported for Intercepted Packets 97
  High Availability for Lawful Intercept 97
    Preserving TAP and MD Table during RP Fail Over 97
    Replay Timer 97

CHAPTER 9
Restrictions for IEEE 802.1X Port-Based Authentication 99
  IEEE 802.1X Device Roles 99
  Understanding 802.1X Port-Based Authentication 99
Configure 802.1X Port-Based Authentication 100
  Configure RADIUS Server 101
  Configure 802.1X Authentication Method 101
Create 802.1X Profile 101
  Apply 802.1X Profile on Interface 101
Verify 802.1X Port-Based Authentication 102
  Show Command Outputs 102
  Syslog Messages 103
Configuring AAA Services

This module describes the implementation of the administrative model of task-based authorization used to control user access in the 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 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 software base package and is available by default.

- Configuring AAA Services, on page 1

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 the Cisco 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 32” section.

Configure Task group

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.

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.

Procedure

Step 1 configure

Step 2 taskgroup taskgroup-name

Example:

RP/0/RP0/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:

RP/0/RP0/CPU0:router(config-tg)# description this is a sample task group description

(Optional) Creates a description of the task group named in Step 2.

Step 4  task {read | write | execute | debug} taskid-name

Example:

RP/0/RP0/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 for each task ID to be associated with the task group named in Step 2.

Step 6  commit

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.

Configure 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 owner-sdr.

Procedure

Step 1  configure
Step 2  usergroup  usergroup-name

Example:
RP/0/RP0/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/RP0/CPU0:router(config-ug)#
description this is a sample user group description

(Optional) Creates a description of the user group named in Step 2.

Step 4  inherit usergroup  usergroup-name

Example:
RP/0/RP0/CPU0:router(config-ug)#
inherit usergroup sales

• Explicitly defines permissions for the user group.

Step 5  taskgroup  taskgroup-name

Example:
RP/0/RP0/CPU0:router(config-ug)# taskgroup beta

Associates the user group named in Step 2 with 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 6  Repeat Step for each task group to be associated with the user group named in Step 2.

—

Step 7  commit

---

Configure Users

Perform this task to configure a user.

Each user is identified 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.

Procedure

Step 1  configure
Step 2  username  user-name
Example:
RP/0/RP0/CPU0:router(config)# username user1

Creates a name for a new user (or identifies a current user) and enters username configuration submode.

• The \textit{user-name} argument can be only one word. Spaces and quotation marks are not allowed.

\textbf{Step 3}

Do one of the following:

• \texttt{password \{0 | 7\} password}
• \texttt{secret \{0 | 5\} secret}

\textbf{Example:}
RP/0/RP0/CPU0:router(config-un)# password 0 pwd1
or
RP/0/RP0/CPU0:router(config-un)# secret 0 sec1

Specifies a password for the user named in step 2.

• Use the \texttt{secret} command to create a secure login password for the user names specified in step 2.

• Entering \texttt{0} following the \texttt{password} command specifies that an unencrypted (clear-text) password follows. Entering \texttt{7} following the \texttt{password} command specifies that an encrypted password follows.

• Entering \texttt{0} following the \texttt{secret} command specifies that a secure unencrypted (clear-text) password follows. Entering \texttt{5} following the \texttt{secret} command specifies that a secure encrypted password follows.

• Type \texttt{0} is the default for the \texttt{password} and \texttt{secret} commands.

\textbf{Step 4}

\texttt{group group-name}

\textbf{Example:}
RP/0/RP0/CPU0:router(config-un)# group sysadmin

Assigns the user named in step 2 to a user group that has already been defined through the \texttt{usergroup} command.

• 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.

\textbf{Step 5}

Repeat step 4 for each user group to be associated with the user specified in step 2.

\textbf{Step 6}

\texttt{commit}

---

\textbf{Configure 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
Configure Router to RADIUS Server Communication

- 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.

---

**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.

---

**Procedure**

1. **configure**
2. `radius-server host {hostname | ip-address} [auth-port port-number] [acct-port port-number] [timeout seconds] [retransmit retries] [key string]

Example:

```plaintext
RP/0/RP0/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 the encryption key used on the RADIUS server. Always configure the key as the last item in the `radius-server host` command syntax because the leading spaces are ignored, but spaces within and at the end of the key are used. If you use spaces in your key, do not enclose the key in quotation marks unless the quotation marks themselves are part of the key.

Step 3 `radius-server retransmit` `retries`

Example:
```
RP/0/RP0/CPU0:router(config)# radius-server retransmit 5
```

Specifies the number of times the 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:
```
RP/0/RP0/CPU0:router(config)# radius-server timeout 10
```

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.

Step 5 `radius-server key` `{0 clear-text-key | 7 encrypted-key | clear-text-key}`

Example:
```
RP/0/RP0/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/RP0/CPU0:router(config)# radius source-interface 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 through step 6 for each external server to be configured.
Configure RADIUS Dead-Server Detection

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.

**Procedure**

1. **Step 1** configure
2. **Step 2** radius-server deadtime `minutes`
   
   **Example:**
   
   ```
   RP/0/RP0/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.

3. **Step 3** radius-server dead-criteria time `seconds`
   
   **Example:**
   
   ```
   RP/0/RP0/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.

4. **Step 4** radius-server dead-criteria tries `tries`
   
   **Example:**
   
   ```
   RP/0/RP0/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.

5. **Step 5** commit
6. **Step 6** show radius dead-criteria host `ip-addr [auth-port auth-port] [acct-port acct-port]`
   
   **Example:**
   
   ```
   RP/0/RP0/CPU0:router# show radius dead-criteria host 172.19.192.80
   ```
Configure 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.

**Procedure**

**Step 1**
```
configure
```

**Step 2**
```
tacacs-server host host-name port port-number
```
*Example:*
```
RP/0/RP0/CPU0:router(config)# tacacs-server host 209.165.200.226 port 51
RP/0/RP0/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/RP0/CPU0:router(config-tacacs-host)# tacacs-server host 209.165.200.226 timeout 30
RP/0/RP0/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/RP0/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.
Configure TACACS+ Server

- (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/RP0/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 source-interface type instance`

*Example:*

```
RP/0/RP0/CPU0:router(config)# tacacs source-interface 0/4/0/0
```

(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 7** Repeat step 2 through step 5 for each external server to be configured.

---

**Step 8** `commit`

**Step 9** `show tacacs`

*Example:*

```
RP/0/RP0/CPU0:router# show tacacs
```

(Optional) Displays information about the TACACS+ servers that are configured in the system.

---

**Tacacs Summary Example:**

```
! OOB TAC
tacacs-server host 123.100.100.186 port 49
    key lm51
    !
tacacs-server host 123.100.100.187 port 49
    key lm51
    !
    aaa group server tacacs+ tacgrp
    server 123.100.100.186
    server 123.100.100.187
    !
    aaa group server tacacs+ eem
```
server 123.100.100.186
server 123.100.100.187
!
aaa authorization exec tacauthen group tacgrp local
aaa authentication login taclogin group tacgrp local
!
line console
authorization exec tacauthen
login authentication taclogin
timeout login response 30
timestamp
eexec-timeout 0 0
session-timeout 15
!
vty-pool default 0 99 line-template console

Configure 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).

**Before you begin**

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

**Procedure**

**Step 1**

configure

**Step 2**

`aaa group server radius group-name`

Example:

```
RP/0/RP0/CPU0:router(config)# aaa group server radius radgroup1
```

Groups different server hosts into distinct lists and enters the server group configuration mode.

**Step 3**

`server {hostname | ip-address} [auth-port port-number] [acct-port port-number]`

Example:

```
RP/0/RP0/CPU0:router(config-sg-radius)# server 192.168.20.0
```

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 for every external server to be added to the server group named in step 3.

**Step 5**

`deadtime minutes`

Example:

```
RP/0/RP0/CPU0:router(config-sg-radius)# deadtime 1
```
Configure 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 deadtime after the group is created.

---

**Step 6**
commit

**Step 7**
show radius server-groups [group-name [detail]]

**Example:**
RP/0/RP0/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.

---

**Configure 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).

**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.

**Procedure**

**Step 1**
configure

**Step 2**
aaa group server tacacs+ group-name

**Example:**
RP/0/RP0/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}

**Example:**
RP/0/RP0/CPU0:router(config-sg-tacacs+)# server 192.168.100.0

Specifies the hostname or IP address of an external TACACS+ server.
• When configured, this group can be referenced from the AAA method lists (used while configuring authentication, authorization, or accounting).

Step 4  (Optional) vrf vrf-id

Example:
RP/0/RP0/CPU0:router(config-sg-tacacs+)# vrf vrf-id

The vrf option specifies the Virtual Private Network (VPN) routing and forwarding (VRF) reference of an AAA TACACS+ server group.

Step 5  Repeat step 3 for every external server to be added to the server group named in step 2.

Step 6  (Optional) vrf vrf-id

The vrf option specifies the Virtual Private Network (VPN) routing and forwarding (VRF) reference of an AAA TACACS+ server group.

Step 7  commit

Step 8  show tacacs server-groups

Example:
RP/0/RP0/CPU0:router# show tacacs server-groups

(Optional) Displays information about each TACACS+ server group that is configured in the system.

Create Series of Authentication Methods

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 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. 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

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.

Procedure

**Step 1** configure

**Step 2** `aaa authentication {login} {default | list-name} method-list`

Example:

```
RP/0/RP0/CPU0#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 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
- The example specifies the `default` method list to be used for authentication.

**Step 3** `commit`

**Step 4** Repeat Step 1 through Step 3 for every authentication method list to be configured.
Create Series of Authorization Methods

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.

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.

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 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.

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.

**Procedure**

1. configure
2. aaa authorization {commands | eventmanager | exec | network} {default | list-name} {none | local | group {tacacs+ | radius | group-name}}

**Example:**

```
RP/0/RP0/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 XR EXEC mode shell commands. Command authorization applies to the EXEC mode commands issued by a user. Command authorization attempts authorization for all XR EXEC mode commands.
• The **eventmanager** keyword applies an authorization method for authorizing an event manager (fault manager).

• The **exec** keyword configures authorization for an interactive (XR EXEC mode) session.

• The **network** keyword configures authorization for network services like PPP or IKE.

• 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**  
**commit**

---

**Create 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.

Currently, the 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.
Procedure

Step 1 configure

Step 2 Do one of the following:

- aaa accounting {commands | exec | network} {default | list-name} {start-stop | stop-only}
- {none | method}

Example:

RP/0/RP0/CPU0:router(config)# aaa accounting commands default stop-only group tacacs+

Note Command accounting is not supported on RADIUS, but supported on TACACS.

Creates a series of accounting methods, or a method list.

- The **commands** keyword enables accounting for XR EXEC mode shell commands.
- The **exec** keyword enables accounting for an interactive (XR 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.
  - **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 commit
Generate 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, 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.

Procedure

Step 1

configure

Step 2

`aaa accounting update {newinfo | periodic minutes}`

Example:

```
RP/0/RP0/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 congestion when many users are logged in to the network.

Step 3

commit

Apply Method List

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.

Procedure

Step 1

configure

Step 2

`line { console | default | template template-name}`

Example:
Enable Accounting Services

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

Procedure

Step 1 configure
Step 2 line { console | default | template template-name}
Example:

RP/0/RP0/CPU0:router(config)# line console

Enters line template configuration mode.

Step 3 accounting {commands | exec} {default | list-name}
Example:

RP/0/RP0/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 XR EXEC mode shell commands.

The **exec** keyword enables accounting for an interactive (XR 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** commit

---

**What to do next**

After applying accounting method lists by enabling AAA accounting services, configure login parameters.

### Configure Login Parameters

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

**Procedure**

**Step 1** configure

**Step 2** line template  *template-name*

**Example:**

```
RP/0/RP0/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/RP0/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** commit

---

### 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:

\[
\text{task} = \text{"permissions : taskid name , # usergroup name , ..."}
\]

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:

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:**

```plaintext
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 XR EXEC mode to display all the tasks user1 can perform. For example:

**Example:**

```
Username:user1
Password:
RP/0/RP0/CPU0:router# show user tasks
```

<table>
<thead>
<tr>
<th>Task</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic-services</td>
<td>READ, WRITE, EXECUTE, DEBUG</td>
</tr>
<tr>
<td>bgp</td>
<td>READ, WRITE</td>
</tr>
<tr>
<td>cdp</td>
<td>READ</td>
</tr>
<tr>
<td>diag</td>
<td>READ</td>
</tr>
<tr>
<td>ext-access</td>
<td>READ</td>
</tr>
<tr>
<td>logging</td>
<td>READ</td>
</tr>
</tbody>
</table>

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/RP0/CPU0:router# show user tasks
No task ids available
```

---

**References for AAA Services**

This section lists all the conceptual information that a 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**

User attributes form the basis of the Cisco 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).
User Categories

Router users are classified into the following categories:

- 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).
- 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).
- 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.
- 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.

Secure Domain Router (SDR) Users

A SDR user has restricted access to an SDR as determined by the 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. Multiple SDRs in a chasis are not supported.

User Groups

A user group defines a collection of users that share a set of attributes, such as access privileges. Cisco 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.

• Each user task can be assigned read, write, execute, or debug permission.

Predefined User Groups

The Cisco 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.

• **netadmin**: Has the ability to control and monitor all system and network parameters.

• **operator**: A demonstration group with basic privileges.

• **root-lr**: Has the ability to control and monitor the specific secure domain router.

• **sysadmin**: Has the ability to control and monitor all system parameters but cannot configure network protocols.

• **serviceadmin**: Service administration tasks, for example, Session Border Controller (SBC).

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**

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.

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

• **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. 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.

Command Access in XR and Admin Modes

The XR user group and task is mapped to the System Admin VM group when the System Admin mode is accessed from XR mode using `admin` command. The corresponding access permission on System Admin VM is available to the user. Currently, only aaa, admin task and root-lr groups are mapped to System Admin VM group or task. The other tasks like protocols are not mapped as these services are not supported in System Admin VM. The disaster-recovery user of System Admin VM is synced with the Host VM.

<table>
<thead>
<tr>
<th>XR Task or Group</th>
<th>Sysadmin VM Group</th>
<th>Access</th>
<th>Example</th>
</tr>
</thead>
</table>
| root-lr          | Root-system group | Full access to the system configuration. | RP/0/RP0/CPU0:ios#show user group
Mon Nov 3 13:48:54.536 UTC
root-lr, cisco-support
RP/0/RP0/CPU0:ios#show user tasks | inc
root-lr
Mon Nov 3 13:49:06.495 UTC
Task: root-lr : READ WRITE
EXECUTE DEBUG (reserved)
RP/0/RP0/CPU0:ios#admin
sysadmin-vm:0_RP0# show aaa user-group
Mon Nov 3 13:48:00.790 UTC
User group : root-system |

| Admin-r/w/x/d     | Admin-r           | Read only commands on Sysadmin VM | taskgroup tg-admin-write
| Task: admin : READ WRITE EXECUTE |

<p>| root-lr | Read only commands on Sysadmin VM | admin-write connected from 127.0.0.1 using console on xr-vm_node0_RP0_CPU0 | sysadmin-vm:0_RP0# show aaa user-group Mon Nov 3 13:53:00.790 UTC User group : admin-r |</p>
<table>
<thead>
<tr>
<th>XR Task or Group</th>
<th>Sysadmin VM Group</th>
<th>Access</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Netadmin or sysadmin group</td>
<td>Aaa-r and admin -r</td>
<td>Read only commands on Sysadmin VM</td>
<td>RP/0/RP0/CPU0:ios#show user group Mon Nov 3 13:44:39.176 UTC netadmin RP/0/RP0/CPU0:ios#show user tasks</td>
</tr>
</tbody>
</table>

### 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.

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.

- 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.

- Configuring command authorization or XR 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 one or both of the following:

- Before turning on TACACS+ command authorization or XR 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 XR 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.

**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.

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 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)

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 XR EXEC mode).

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 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.

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, 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 SDR user, or SDR 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 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 XR EXEC mode and external API clients can use this feature to optimize their operations. XR 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.

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.

**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 software by a task-based authorization system.

**Task IDs**

The operational tasks that enable users to control, configure, and monitor Cisco 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, Netconf, Restconf, 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 cannot 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.

**Note**

Restconf will be supported in a future release.

### Table 1: Task ID Classes

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read</td>
<td>Specifies a designation that permits only a read operation.</td>
</tr>
<tr>
<td>Write</td>
<td>Specifies a designation that permits a change operation and implicitly allows a read operation.</td>
</tr>
<tr>
<td>Execute</td>
<td>Specifies a designation that permits an access operation; for example ping and Telnet.</td>
</tr>
<tr>
<td>Debug</td>
<td>Specifies a designation that permits a debug operation.</td>
</tr>
</tbody>
</table>

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/RP0/CPU0:router# show redundancy
```

### Task IDs for TACACS+ and RADIUS Authenticated Users

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

### 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 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`. 
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

Netconf and Restconf for AAA Services

Just as in XML schemas, in Netconf and Restconf, username and password is controlled by either local or triple A (AAA) services.

Note

Restconf will be supported in a future release.

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.
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:
  - 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 another if that router uses 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:
   1. ACCEPT—The user is authenticated.
   1. REJECT—The user is not authenticated and is prompted to reenter the username and password, or access is denied.
   1. CHALLENGE—A challenge is issued by the RADIUS server. The challenge collects additional data from the user.
1. 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 XR 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 XR EXEC mode services.
- Connection parameters, including the host or client IP address, access list, and user timeouts.
CA interoperability permits 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 will be supported in a future release.

- Implementing Certification Authority Interoperability, on page 37

## Implementing Certification Authority Interoperability

CA interoperability permits 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 will be supported in a future release.

## 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 must install and activate the Package Installation Envelope (PIE) for the security software.

  For detailed information about optional PIE installation, refer to the *System Management Guide*.

- 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

Configure 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.

Procedure

Step 1 configure
Step 2 hostname name
Example:

```
RP/0/RP0/CPU0:router(config)# hostname myhost
```

Configures the hostname of the router.

Step 3 domain name domain-name
Example:

```
RP/0/RP0/CPU0:router(config)# domain name mydomain.com
```

Configures the IP domain name of the router.

Step 4 commit

Generate RSA Key Pair

This task generates an RSA key pair.

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 crypto key generate rsa [usage keys | general-keys] [keypair-label]
Example:

```
RP/0/RP0/CPU0:router# crypto key generate rsa general-keys
```

Generates RSA key pairs.
• 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  
crypto key zeroize rsa [keypair-label]

**Example:**

```
RP/0/RP0/CPU0:router# crypto key zeroize rsa key1
```

(Optional) Deletes all RSAs from the router.

• Under certain circumstances, you may want to delete all RSA keys from your 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  
show crypto key mypubkey rsa

**Example:**

```
RP/0/RP0/CPU0:router# show crypto key mypubkey rsa
```

(Optional) Displays the RSA public keys for your router.

---

## Import 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.

### Procedure

### Step 1  
crypto key import authentication rsa [usage keys | general-keys] [keypair-label]

**Example:**

```
RP/0/RP0/CPU0:router# crypto key import authentication rsa general-keys
```

Generates RSA key pairs.

• 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/RP0/CPU0:router# show crypto key mypubkey rsa
```
Declare Certification Authority and Configure Trusted Point

This task declares a CA and configures a trusted point.

Procedure

Step 1 configure

Step 2 crypto ca trustpoint ca-name

Example:

```
RP/0/RP0/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 3 enrollment url CA-URL

Example:

```
RP/0/RP0/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 4 query url LDAP-URL

Example:

```
RP/0/RP0/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 5 enrollment retry period minutes

Example:

```
RP/0/RP0/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 6  enrollment retry count number

Example:

RP/0/RP0/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 7  rsakeypair keypair-label

Example:

RP/0/RP0/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 8  commit

---

**Authenticate 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.

**Procedure**

Step 1  crypto ca authenticate ca-name

Example:

RP/0/RP0/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/RP0/CPU0:router# show crypto ca certificates

(Optional) Displays information about the CA certificate.
Request 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.

Procedure

Step 1  
crypto ca enroll ca-name

Example:

```
RP/0/RP0/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/RP0/CPU0:router# show crypto ca certificates
```

(Optional) Displays information about the CA certificate.

Configure 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.

Procedure

Step 1  
configure

Step 2  
crypto ca trustpoint  ca-name

Example:

```
RP/0/RP0/CPU0:router(config)# crypto ca trustpoint myca
RP/0//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
 enrolment terminal

**Example:**

```
RP/0/RP0/CPU0:router(config-trustp)# enrollment terminal
```

Specifies manual cut-and-paste certificate enrollment.

### Step 4
 commit

### Step 5
 `crypto ca authenticate ca-name`

**Example:**

```
RP/0/RP0/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.

### Step 6
 `crypto ca enroll ca-name`

**Example:**

```
RP/0/RP0/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/RP0/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.

**Note** You must enter the `crypto ca import` command twice if usage keys (signature and encryption keys) are used. The first time the command is entered, one of the certificates is pasted into the router; the second time the command is entered, the other certificate is pasted into the router. (It does not matter which certificate is pasted first.

### Step 8
 `show crypto ca certificates`

**Example:**

```
RP/0/RP0/CPU0:router# show crypto ca certificates
```
Displays information about your certificate and the CA certificate.

The following examples show 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 OD6092A 86486F7 0D010101 05000381 8D003081 890281B8 07A417FD 7979D262 B35465A6 1D5B70D1 36ACAFBD 7F91D5A0 CFB0EE91 B9D52C69

! 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
```

Do you accept this certificate?? [yes/no]: yes

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

Certificate Authority Trust Pool Management

The trust pool feature is used to authenticate sessions, such as HTTPS, that occur between devices by using commonly recognized trusted agents called certificate authorities (CAs). This feature is enabled by default in the software to create a scheme to provision, store, and manage a pool of certificates from known CAs in a way similar to the services a browser provides for securing sessions. A special trusted point called a trust pool is designated, containing multiple known CA certificates from Cisco and possibly from other vendors. The trust pool consists of both built-in and downloaded CA certificates.

Implementing Certification Authority Interoperability provides details on Certificate Authority and trusted point.
CA Certificate Bundling in the Trust Pool

The router uses a built-in CA certificate bundle that is packaged into the asr9k-k9sec PIE. The bundle is contained in a special certificate store called a CA trust pool, which is updated automatically by Cisco. This trust pool is known by Cisco and other vendors. A CA certificate bundle can be in the following formats:

- Privilege Management Infrastructure (PMI) certificates in Distinguished Encoding Rules (DER) binary format enveloped within a public-key cryptographic message syntax standard 7 (pkcs7).
- A file containing concatenated X.509 certificates in Privacy Enhanced Mail (PEM) format with PEM headers.

Prerequisites for CA Trust Pool Management

Restrictions for CA trust pool management

Updating the CA Trustpool

The CA trustpool must be updated when the following conditions occur:

- A certificate in the trustpool is due to expire or has been reissued.
- The published CA certificate bundle contains additional trusted certificates that are needed by a given application.
- The configuration has been corrupted.

The CA trustpool is considered as a single entity, As such, any update you perform will replace the entire trustpool.

Note

A built-in certificate in the trustpool cannot be physically replaced. However, a built-in certificate is rendered inactive after an update if its X.509 subject-name attribute matches the certificate in the CA certificate bundle.

Following are the methods available for updating the certificates in the trustpool:

- **Automatic update**: A timer is established for the trustpool that matches the CA certificate with the earliest expiration time. If the timer is running and a bundle location is not configured and not explicitly disabled, syslog warnings should be issued at reasonable intervals to alert the admin that this trustpool policy option is not set. Automatic trustpool updates use the configured URL. When the CA trustpool expires, the policy is read, the bundle is loaded, and the PKI trustpool is replaced. If the automatic CA trustpool update encounters problems when initiating, then the following schedule is used to initiate the update until the download is successful: 20 days, 15 days, 10 days, 5 days, 4 days, 3 days, 2 days, 1 day, and then once every hour.

- **Manual update**: Manually Update Certificates in Trust Pool, on page 46 provides details.

Manually Update Certificates in Trust Pool

The CA trust pool feature is enabled by default and uses the built-in CA certificate bundle in the trust pool, which receives automatic updates from Cisco. Perform this task to manually update certificates in the trust pool if they are not current, are corrupt, or if certain certificates need to be updated.
### Manually Update Certificates in Trust Pool

#### Procedure

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td><strong>crypto ca trustpool import url clean</strong>&lt;br&gt;<strong>Example:</strong>&lt;br&gt;RP/0/RSP0RP0/CPU0:IMC0#crypto ca trustpool import url clean</td>
<td><em>(Optional)</em> Manually removes all downloaded CA certificates. This command is run in the EXEC mode.</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td><strong>crypto ca trustpool import url</strong>&lt;br&gt;<strong>Example:</strong>&lt;br&gt;RP/0/RSP0RP0/CPU0:IMC0#crypto ca trustpool import url <a href="http://www.cisco.com/security/pki/trs/ios.p7b">http://www.cisco.com/security/pki/trs/ios.p7b</a></td>
<td>Specify the URL from which the CA trust pool certificate bundle must be downloaded. This manually imports (downloads) the CA certificate bundle into the CA trust pool to update or replace the existing CA certificate bundle.</td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td><strong>show crypto ca trustpool policy</strong>&lt;br&gt;<strong>Example:</strong>&lt;br&gt;RP/0/RSP0RP0/CPU0:IMC0#show crypto ca trustpool&lt;br&gt;Trustpool: Built-In</td>
<td>Displays the CA trust pool certificates of the router in a verbose format.</td>
</tr>
</tbody>
</table>
## Configuring Optional Trustpool Policy Parameters

### Procedure

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>configure</td>
<td>Enters ca-trustpool configuration mode where commands can be accessed to configure CA trustpool policy parameters.</td>
</tr>
<tr>
<td>2</td>
<td>crypto ca trustpool policy</td>
<td>Enters ca-trustpool configuration mode where commands can be accessed to configure CA trustpool policy parameters.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>crypto ca trustpool policy</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>RP/0/RSP0/CPU0:IMC0(config)#crypto ca trustpool policy</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>RP/0/RSP0/CPU0:IMC0(config-trustpool)#</code></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>cabundle url URL</td>
<td>Specifies the URL from which the CA trustpool certificate bundle is downloaded.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>cabundle url</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>RP/0/RSP0/CPU0:IMC0(config-trustpool)#cabundle url</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>http://www.cisco.com/security/pki/crl/crca2048.crl</code></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>crl optional</td>
<td>Disables revocation checking when the trustpool policy is being used. By default, the router enforces a check of the revocation status of the certificate by querying the certificate revocation list (CRL).</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>crl optional</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>RP/0/RSP0/CPU0:IMC0(config-trustpool)#crl optional</code></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>description LINE</td>
<td>Specifies the URL from which the CA trustpool certificate bundle is downloaded.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>description</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>RP/0/RSP0/CPU0:IMC0(config-trustpool)#description</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>Trustpool for Test</code></td>
<td></td>
</tr>
</tbody>
</table>

### Handling of CA Certificates appearing both in Trust Pool and Trust Point

There may be cases where a CA resides in both the trust pool and a trust point; for example, a trust point is using a CA and a CA bundle is downloaded later with this same CA inside. In this scenario, the CA in the trust point and its policy is considered, before the CA in the trust pool or trust pool policy to ensure that any current behavior is not altered when the trust pool feature is implemented on the router.

The policy indicates how the security appliance obtains the CA certificate and the authentication policies for user certificates issued by the CA.

## Information About Implementing Certification Authority

### Supported Standards for Certification Authority Interoperability

Cisco supports the following standards:

- **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 (SA).
• 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

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.
CA Registration Authorities
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.

- Implementing Keychain Management, on page 51

Restrictions for Implementing Keychain Management

You must be aware that changing the system clock impacts the validity of the keys in the existing configuration.

Configure Keychain

This task configures a name for the keychain.

You can create or modify the name of the keychain.

Procedure

| Step 1 | configure |
| Step 2 | key chain key-chain-name |

Example:

```
RP/0/RP0/CPU0:router(config)# key chain isis-keys
RP/0/RP0/CPU0:router(config-isis-keys)#
```

Creates a name for the keychain.
Configuring only the keychain name without any key identifiers is considered a nonoperation. When you exit the configuration, the router does not prompt you to commit changes until you have configured the key identifier and at least one of the mode attributes or keychain-key configuration mode attributes (for example, lifetime or key string).

Step 3  commit
Step 4  show key chain key-chain-name

Example:

RP/0/RP0/CPU0:router# show key chain isis-keys

(Optional) Displays the name of the keychain.

Note  The key-chain-name argument is optional. If you do not specify a name for the key-chain-name argument, all the keychains are displayed.

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
end

Uncommitted changes found, commit them? [yes]: yes

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]
```

**Configure 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.

**Procedure**

Step 1  configure
Step 2  
key chain *key-chain-name*

Example:

```
RP/0/RP0/CPU0:router(config)# key chain isis-keys
```

Creates a name for the keychain.

Step 3  
accept-tolerance *value* [infinite]

Example:

```
RP/0/RP0/CPU0:router(config-isis-keys)# accept-tolerance infinite
```

Configures a tolerance value to accept keys for the keychain.

- 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 the tolerance specification is infinite.

Step 4  
commit

---

**Configure Key Identifier for Keychain**

This task configures a key identifier for the keychain.

You can create or modify the key for the keychain.

**Procedure**

Step 1  
configure

Step 2  
key chain *key-chain-name*

Example:

```
RP/0/RP0/CPU0:router(config)# key chain isis-keys
```

Creates a name for the keychain.

Step 3  
key *key-id*

Example:

```
RP/0/RP0/CPU0:router(config-isis-keys)# key 8
```

Creates a key for the keychain. The key ID number is translated from decimal to hexadecimal to create the command mode subprompt.

- Use the *key-id* argument as a 48-bit integer.

Step 4  
commit
Configure Text for Key String

This task configures the text for the key string.

Procedure

Step 1 configure

Step 2 key chain key-chain-name

Example:

RP/0/RP0/CPU0:router(config)# key chain isis-keys

Creates a name for the keychain.

Step 3 key key-id

Example:

RP/0/RP0/CPU0:router(config-isis-keys)# key 8
RP/0/RP0/CPU0:router(config-isis-keys-0x8)#

Creates a key for the keychain.

Step 4 key-string [clear | password] key-string-text

Example:

RP/0/RP0/CPU0:router(config-isis-keys-0x8)# key-string password 8

Specifies the text string for the key.

• Use the clear keyword to specify the key string in clear text form; use the password keyword to specify the key in encrypted form.

Step 5 commit

Determine Valid Keys

This task determines the valid keys for local applications to authenticate the remote peers.

Procedure

Step 1 configure

Step 2 key chain key-chain-name

Example:

RP/0/RP0/CPU0:router(config)# key chain isis-keys

Creates a name for the keychain.
Step 3  key key-id  
Example:  

RP/0/RP0/CPU0:router(config-isis-keys)# key 8  
RP/0/RP0/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/RP0/CPU0:router(config-isis-keys)# key 8  
RP/0/RP0/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.  

Step 5  commit  

---  

**Configure Keys to Generate Authentication Digest for Outbound Application Traffic**  

This task configures the keys to generate authentication digest for the outbound application traffic.  

**Procedure**  

Step 1  configure  
Step 2  key chain key-chain-name  
Example:  

RP/0/RP0/CPU0:router(config)# key chain isis-keys  

Creates a name for the keychain.  

Step 3  key key-id  
Example:  

RP/0/RP0/CPU0:router(config-isis-keys)# key 8  
RP/0/RP0/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/RP0/CPU0:router(config-isis-keys)#key 8  
RP/0/RP0/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

If you intend to set lifetimes on keys, Network Time Protocol (NTP) or some other time synchronization method is recommended.

**Step 5**  
**commit**

---

**Configure Cryptographic Algorithm**

This task allows the keychain configuration to accept the choice of the cryptographic algorithm.

**Procedure**

**Step 1**  
**configure**

**Step 2**  
**key chain** *key-chain-name*

**Example:**

```
RP/0/RP0/CPU0:router(config)# key chain isis-keys
RP/0/RP0/CPU0:router(config-isis-keys)#
```

Creates a name for the keychain.

**Step 3**  
**key** *key-id*

**Example:**

```
RP/0/RP0/CPU0:router(config-isis-keys)# key 8
RP/0/RP0/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/RP0/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
The routing protocols each support a different set of cryptographic algorithms:

- Border Gateway Protocol (BGP) supports only HMAC-MD5 and HMAC-SHA1-12.
- Open Shortest Path First (OSPF) supports MD5, HMAC-MD5, HMAC-SHA-256, HMAC-SHA1-12, HMAC-SHA1-20, and HMAC-SHA1-96.

### 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.

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.
Implementing Keychain Management

Lifetime of Key
Implementing URPF

This section describes the implementation of URPF.

- Understanding URPF, on page 59
- Configuring URPF Loose Mode, on page 59

Understanding URPF

It has become a commonplace practice for hackers planning a DoS attack to use forged IP addresses (the practice is known as IP address spoofing) and constantly change the source IP address to avoid detection by service providers.

Unicast Reverse Path Forwarding (URPF) is a mechanism for validating the source IP address of packets received on a router. A router configured with URPF performs a reverse path lookup in the FIB table to validate the presence of the source IP address. If the source IP address is listed in the table, then it indicates that the source is reachable and valid. If source IP address cannot be located in the FIB table, the packet is treated as malicious by the router and discarded.

The router supports the use of URPF in loose mode. URPF loose mode is enabled when the router is configured to validate only the prefix of the source IP address in the FIB and not the interface used by the packet to reach the router. By configuring loose mode, legitimate traffic that uses an alternate interface to reach the router is not mistaken to be malicious. URPF loose mode is very useful in multi-homed provider edge networks.

Configuring URPF Loose Mode

This section explains how you can configure URPF loose mode on the router for both IPv4 and IPv6 networks.

Before You Begin

Before you can configure URPF loose mode on a router, you must disable the default scale on the line card, as described in this section.

Note

IPv6 uRPF configuration requires the `hw-module fib ipv6 scale internet-optimized-disable` command for all types of cards, both TCAM cards and non-TCAM cards. By default, IPv6 uses internal memory for prefixes. Therefore, you need to configure the `hw-module fib ipv6 scale internet-optimized-disable` command and then reload the line card.
Configuring URPF Loose Mode

**Note**
Line cards must be reloaded after disabling the default scale. This is done to ensure that the `hw-module` command configuration takes immediate effect.

**Note**
On Jericho+ with external TCAM (eTCAM) systems, the dual capacity mode need not be disabled to enable uRPF.

**For all types of line cards with TCAM:**

```
RP/0/RP0/CPU0:router(config)# hw-module tcam fib ipv4 scaledisable
RP/0/RP0/CPU0:router(config)# hw-module fib ipv6 scale internet-optimized-disable
RP/0/RP0/CPU0:router(config)# commit
RP/0/RP0/CPU0:router(config)# end
RP/0/RP0/CPU0:router# reload location all
Proceed with reload? [confirm]
```

**For all types of line cards without TCAM:**

```
RP/0/RP0/CPU0:router(config)# hw-module fib ipv4 scale host-optimized-disable
RP/0/RP0/CPU0:router(config)# hw-module fib ipv6 scale internet-optimized-disable
RP/0/RP0/CPU0:router(config)# commit
RP/0/RP0/CPU0:router(config)# end
RP/0/RP0/CPU0:router# reload location all
Proceed with reload? [confirm]
```

Configuration

Use the following configuration to configure URPF loose mode on the router.

**Note**
You must configure both IPv4 and IPv6 commands (as described in this section) for URPF to work.

```
RP/0/RP0/CPU0:router(config)# interface bundle-ether1
RP/0/RP0/CPU0:router(config-if)# ipv4 address 10.0.0.1 255.255.255.0
RP/0/RP0/CPU0:router(config-if)# ipv4 verify unicast source reachable-via any
RP/0/RP0/CPU0:router(config-if)# ipv6 address 2001::1/64
RP/0/RP0/CPU0:router(config-if)# ipv6 verify unicast source reachable-via any
RP/0/RP0/CPU0:router(config-if)# commit
```

Running Configuration

Confirm your configuration as shown:

```
RP/0/RP0/CPU0:router(config-if)# show running-config
Thu Jul 27 14:40:38.167 IST
...
interface Bundle-Ether1
 ipv4 address 10.0.0.1 255.255.255.0
 ipv4 verify unicast source reachable-via any
ipv6 address 2001::1/64
 ipv6 verify unicast source reachable-via any
```
You have successfully configured URPF loose mode on the router.
Implementing Management Plane Protection

The Management Plane Protection (MPP) feature 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.

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.

Afterwards, only the default management interfaces and those you have previously configured as MPP interfaces accept network management packets destined for the device. All other interfaces drop such packets. Logical interfaces (or any other interfaces not present on the data plane) filter packets based on the ingress physical interface.

- Implementing Management Plane Protection, on page 63

Implementing Management Plane Protection

The Management Plane Protection (MPP) feature 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.

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.

Afterwards, only the default management interfaces and those you have previously configured as MPP interfaces accept network management packets destined for the device. All other interfaces drop such packets. Logical interfaces (or any other interfaces not present on the data plane) filter packets based on the ingress physical interface.
Benefits of Management Plane Protection

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.

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.
- 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), XML and Netconf.
- MPP does not support MIB.

Configure Device for Management Plane Protection for Inband Interface

An inband management interface is a 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. 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.
Implementing Management Plane Protection

Configure Device for Management Plane Protection for Inband Interface

**Procedure**

<table>
<thead>
<tr>
<th>Step 1</th>
<th>configure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 2</td>
<td>control-plane</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RP0/CPU0:router(config)# control-plane
RP/0/RP0/CPU0:router(config-ctrl)#
```

Enters control plane configuration mode.

| Step 3 | management-plane |

**Example:**

```
RP/0/RP0/CPU0:router(config-ctrl)# management-plane
RP/0/RP0/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/RP0/CPU0:router(config-mpp)# inband
RP/0/RP0/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/RP0/CPU0:router(config-mpp-inband)# interface HundredGigE 0/6/0/10/0/1/0
RP/0/RP0/CPU0:router(config-mpp-inband-Gi0_0_1_0)#
```

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:**

```
RP/0/RP0/CPU0:router(config-mpp-inband-Gi0_0_1_0)# allow Telnet peer
RP/0/RP0/CPU0:router(config-telnet-peer)#
```
Configure an interface as an inband interface for a specified protocol or all protocols.

- Use the `protocol` argument to allow management protocols on the designated management interface.
  - SNMP (also versions)
  - Secure Shell (v1 and v2)
  - TFTP
  - Telnet
  - Netconf
  - XML

- 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/RP0/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**  
**commit**

**Step 9**  
**show mgmt-plane [inband | ] [interface {type instance}]**

**Example:**

```
RP/0/RP0/CPU0:router# show mgmt-plane inband interface HundredGigE 0/6/0/10/0/1/0
```

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 `interface` keyword to display the details for a specific interface.

---

**Configure Device for Management Plane Protection for Out-of-band 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.

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.

**Procedure**

<table>
<thead>
<tr>
<th>Step 1</th>
<th>configure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 2</td>
<td>control-plane</td>
</tr>
</tbody>
</table>

**Example:**

```
RP/0/RP0/CPU0:router(config)# control-plane
RP/0/RP0/CPU0:router(config-ctrl)#
```

Enters control plane configuration mode.

| Step 3   | management-plane |

**Example:**

```
RP/0/RP0/CPU0:router(config-ctrl)# management-plane
RP/0/RP0/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/RP0/CPU0:router(config-mpp)# out-of-band
RP/0/RP0/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:**
Configure Device for Management Plane Protection for Out-of-band Interface

Step 6

interface {type instance | all}

Example:

```
RP/0/RP0/CPU0:router(config-mpp-outband)# interface HundredGigE 0/6/0/20/0/1/0
RP/0/RP0/CPU0:router(config-mpp-outband-if)#
```

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/RP0/CPU0:router(config-mpp-outband-if)# allow TFTP peer
RP/0/RP0/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
  - Netconf

- 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 8

address ipv6 {peer-ip-address | peer ip-address/length}

Example:
Configure Device for Management Plane Protection for Out-of-band Interface

Step 9
commit

Step 10
show mgmt-plane [inband | out-of-band] [interface {type instance} | vrf]

Example:

RP/0/RP0/CPU0:router# show mgmt-plane out-of-band interface HundredGigE 0/6/0/20/0/1/0

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.

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 HundredGigE 0/6/0/00/0/1/0
    allow all
    allow SSH
    allow Telnet peer
    address ipv4 10.1.0.0/16
!>
    interface HundredGigE 0/6/0/10/0/1/0
    allow Telnet peer
    address ipv4 10.1.0.0/16
!>
```
Information About Implementing Management Plane Protection

Before you enable the Management Plane Protection feature, you should understand the following concepts:

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

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 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, SSH, XML and Netconf. 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.
Implementing Management Plane Protection

Management Plane
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.

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.

It is mandatory to configure address family, protocol, local port, and vrf, as well as at least one of interface or local or remote address.

Note

- gRPC Protocol, on page 73
- Limitations for Traffic Protection for Third-Party Applications, on page 74
- Prerequisites for Traffic Protection for Third-Party Applications Over GRPC, on page 74
- Configuring Traffic Protection for Third-Party Applications, on page 74
- Troubleshooting Traffic Protection for Third-Party Applications, on page 75

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.
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/CP00:ios#configure
RP/0/0/CP00:ios(config)#tpa
RP/0/0/CP00:ios(config-tpa)#vrf default
RP/0/0/CP00:ios(config-tpa-vrf)#address-family ipv4
RP/0/0/CP00:ios(config-tpa-vrf-afi)#protection
RP/0/0/CP00: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
```
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 10.0.1.1/24 local-address 192.168.0.1/32
allow protocol tcp local-port 57600 remote-address 10.0.2.3/24 local-address 192.168.0.1/32

address-family ipv6
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::1/128 local-address 2001:DB8:0:ABCD::1/128
allow protocol tcp local-port 57600 remote-address 2001:DB8::1/128 local-address 2001:DB8:0:ABCD::1/128

!
!

Troubleshooting Traffic Protection for Third-Party Applications

The following show command output verifies whether TPA 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

<table>
<thead>
<tr>
<th>Owner</th>
<th>AF</th>
<th>Proto</th>
<th>State</th>
<th>Interface</th>
<th>VRF</th>
<th>Local ip,port &gt; Remote ip,port</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux</td>
<td>2</td>
<td>6</td>
<td>Prog</td>
<td>global-vrf</td>
<td></td>
<td>any,57600</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>any</td>
<td></td>
<td></td>
<td></td>
<td>any,57600</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Owner</th>
<th>AF</th>
<th>Proto</th>
<th>State</th>
<th>Interface</th>
<th>VRF</th>
<th>Local ip,port &gt; Remote ip,port</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client 2</td>
<td>6 Prog</td>
<td>default</td>
<td>192.168.0.1/32,57600 &gt; 10.0.0.2/32,0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linux 2</td>
<td>6 Ovr</td>
<td>global-vrf</td>
<td>any,57600 &gt; any,0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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# 0/RP0/ADMIN0:Mar 19 15:22:26.837 IST: pm[2433]:
%INFRA-Process_Manager-3-PROCESS_RESTART : Process tams (IID: 0) restarted
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 software supports both SSHv1 and SSHv2.

This module describes how to implement Secure Shell.

• Implementing Secure Shell, on page 77

Prerequisites to Implementing Secure Shell

The following prerequisites are required to implement Secure Shell:

• Download the required image on your router. The SSH server and SSH client require you to have a crypto package (data encryption standard [DES], 3DES and AES) from Cisco downloaded on your router.

• Configure user authentication for local or remote access. You can configure authentication with or without authentication, authorization, and accounting (AAA).

• AAA authentication and authorization must be configured correctly for Secure Shell File Transfer Protocol (SFTP) to work.
Restrictions for Implementing Secure Shell

The following are some basic SSH restrictions and limitations of the SFTP feature:

- 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.

- 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_2.9p2 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 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 onto 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.

Configure SSH

Perform this task to configure SSH.

Note

For SSHv1 configuration, Step 1 to Step 4 are required. For SSHv2 configuration, Step to Step 4 are optional.
**Procedure**

**Step 1**
configure

**Step 2**
hostname *hostname*

Example:

```
RP/0/RP0/CPU0:router(config)# hostname router1
```

Configures a hostname for your router.

**Step 3**
domain name *domain-name*

Example:

```
RP/0/RP0/CPU0:router(config)# domain name cisco.com
```

Defines a default domain name that the software uses to complete unqualified host names.

**Step 4**
commit

**Step 5**
crypto key generate rsa [usage keys | general-keys] [keypair-label]

Example:

```
RP/0/RP0/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/RP0/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/RP0/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**
configure

**Example:**

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

**Step 9**
%s

**Example:**

```
RP/0/RP0/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 aborted.
- If no value is configured, the default value of 30 seconds is used. The range is from 5 to 120.

**Step 10**
Do one of the following:

- `ssh server [vrf vrf-name]`
- `ssh server v2`

**Example:**

```
RP/0/RP0/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.

**Note**

- The SSH server can be configured for multiple VRF usage.
  - (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 11**
commit

**Step 12**
show ssh

**Example:**

```
RP/0/RP0/CPU0:router# show ssh
```

(Optional) Displays all of the incoming and outgoing SSHv1 and SSHv2 connections to the router.

**Step 13**
show ssh session details

**Example:**
Step 14 show ssh history

Example:

```
RP/0/RP0/CPU0:router# show ssh history
```

(Optional) Displays the last hundred SSH connections that were terminated.

Step 15 show ssh history details

Example:

```
RP/0/RP0/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 16 show tech-support ssh

Example:

```
RP/0/RP0/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

---

**Configure SSH Client**

Perform this task to configure an SSH client.

**Procedure**

Step 1 configure

Step 2 ssh client knownhost  device : /filename
Configure SSH Client

Example:

```
RP/0/RP0/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 required. The colon (:) and slash mark (/) are also required.

**Step 3** commit

**Step 4** ssh {ipv4-address | ipv6-address | hostname} [username user- cipher | source-interface type instance]

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, you have not successfully generated an RSA key pair for your router. Make sure that you have specified a hostname and domain. Then use the **crypto key generate rsa** command to generate an RSA key pair and enable the SSH server.
- If you are using SSHv2 and your SSH connection is being rejected, you have not successfully generated a DSA key pair, or a RSA key pair, or an ECDSA key pair for your router. Make sure that you have specified a hostname and domain. Then use the **crypto key generate dsa** command to generate a DSA key pair, or the **crypto key generate rsa** command to generate a RSA key pair, or the **crypto key generate ecdsa [nistp256 | nistp384 | nistp521]** command to generate an ECDSA key pair, and 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.
- 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.
Implementing Secure Shell

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 software authentication. The SSH server in Cisco 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 works with publicly and commercially available SSH servers. The SSH client supports the ciphers of AES, 3DES, message digest algorithm 5 (MD5), SHA1, and password authentication. User authentication is performed in the Telnet session to the router. The user authentication mechanisms supported for SSH are 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)
  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
  PW-Ether PWHE Ethernet Interface
  PW-1M 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
  ```

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 1.1.1.1 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)
```
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 XR 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 connection 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

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.
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.

**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.

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.
Implementing Lawful Intercept

Lawful intercept is the lawfully authorized interception and monitoring of communications of an intercept subject. 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 `ncs5500-li.rpmncs540-li-1.0.0.0-r632361.x86_64.rpm`.

For more information about activating and deactivating the Lawful Intercept package, see the Installing Lawful Intercept (LI) Package, on page 93 section.

| Feature History for Implementing Lawful Intercept |
|------------------------|------------------------|
| **Release** | **Modification** |
| Release 6.3.1 | This feature was introduced. |

- Information About Lawful Intercept Implementation, on page 89
- Prerequisites for Implementing Lawful Intercept, on page 90
- Restrictions for Implementing Lawful Intercept, on page 91
- Lawful Intercept Topology, on page 91
- Benefits of Lawful Intercept, on page 92
- Installing Lawful Intercept (LI) Package, on page 93
- How to Configure SNMPv3 Access for Lawful Intercept, on page 93
- Additional Information on Lawful Intercept, on page 95

Information About Lawful Intercept Implementation

Cisco lawful intercept is based on RFC3924 architecture and SNMPv3 provisioning architecture. SNMPv3 addresses the requirements to authenticate data origin and ensure that the connection from the router to the Mediation Device (MD) is secure. This ensures that unauthorized parties cannot forge an intercept target.

Lawful intercept offers these capabilities:
Prerequisites for Implementing Lawful Intercept

Lawful intercept implementation requires that these prerequisites are met:

- The router is used as content Intercept Access Point (IAP) router in lawful interception operation.

**Provisioned Router**—The router must be already provisioned.

| Tip | For the purpose of lawful intercept taps, provisioning a loopback interface has advantages over other interface types. |

- **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 interface) 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.

- **QoS Peering**—QoS peering must be enabled on the router for Lawful Intercept to work.

| Note | The Lawful Intercept feature has no intersection with the QoS feature on the router. Enabling the QoS peering profile with `hw-module profile qos ingress-model peering` command on all the required line cards, allows QoS and Lawful intercept to allocate hardware resources. |

- 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.

- 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:

- Lawful Intercept shares a pool of 16 unique source IP addresses with tunnel-ip. The combined configuration of GRE tunnel-ips and the MDs (the cTap2MediationSrcInterface field) shall not yield more than 16 unique source IPs. Note that when configuring the MD, if the value 0 is passed in for the cTap2MediationSrcInterface field, it will be resolved into a source IP address, which is the egress IP to the MD destination.

- Lawful intercept is supported only to match pure IP over Ethernet packets.

- Lawful intercept does not provide support for these features on the router:
  - IPv6 multicast tapping
  - Per interface tapping
  - Tagged packet tapping
  - Replicating a single tap to multiple MDs
  - Tapping L2 flows
  - RTP encapsulation

Lawful Intercept Topology

This figure shows intercept access points and interfaces in a lawful intercept topology for both voice and data interception.
Benefits of Lawful Intercept

Lawful intercept has the following benefits:

- Allows multiple LEAs to run a lawful intercept on the same Router without each other’s knowledge.
- Does not affect subscriber services on the router.
- Supports wiretaps in both the input and output direction.
- Supports wiretaps of Layer 3 traffic.
- Cannot be detected by the target.
- Uses Simple Network Management Protocol Version 3 (SNMPv3) and security features such as the View-based Access Control Model (SNMP-VACM-MIB) and User-based Security Model (SNMP-USM-MIB) to restrict access to lawful intercept information and components.
- Hides information about lawful intercepts from all but the most privileged users. An administrator must set up access rights to enable privileged users to access lawful intercept information.
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

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 `ncs5500-li.rpm` `ncs540-li-1.0.0.0-r63236I.x86_64.rpm`.

**Configuration**

```
Router# install add source tftp://223.255.254.252/auto/tftp-sjc-users/username/ncs5500-li-1.0.0.0-r63236I.x86_64.rpm
```

```
Router# install activate ncs5500-li-1.0.0.0-r63236I.x86_64.rpm
```

```
Router# install commit
```

**Verification**

```
Router# show install active
Node 0/RP0/CPU0 [RP]
  Boot Partition: xr_lv0
  Active Packages: 2
  ncs5500-xr-6.3.1.14I version=6.3.1.14I [Boot image]ncs540-xr-6.3.2.36I version=6.3.2.36I [Boot image]
  ncs5500-li-1.0.0.0-r63114Incs540-li-1.0.0.0-r63236I.x86_64.rpm

Node 0/0/CPU0 [LC]
  Boot Partition: xr_lcp_lv0
  Active Packages: 2
  ncs5500-xr-6.3.1.14I version=6.3.1.14I [Boot image]ncs540-xr-6.3.2.36I version=6.3.2.36I [Boot image]
  ncs5500-li-1.0.0.0-r63114Incs540-li-1.0.0.0-r63236I.x86_64.rpm
```

Deactivating the LI RPM

To uninstall the Lawful Intercept package, deactivate `ncs5500-li.rpm` `ncs540-li-1.0.0.0-r63236I.x86_64.rpm` as shown in the following steps:

**Configuration**

```
Router# install deactivate ncs5500-li.rpmncs540-li-1.0.0.0-r63236I.x86_64.rpm
```

```
Router# install commit
```

```
Router# install remove ncs5500-li.rpmncs540-li-1.0.0.0-r63236I.x86_64.rpm
```

```
Router# show install committed
```

How to Configure SNMPv3 Access for Lawful Intercept

Perform these procedures to configure SNMPv3 for the purpose of Lawful Intercept enablement:
Disabling SNMP-based Lawful Intercept

Lawful Intercept is enabled by default on the router after installing and activating the `ncs5500-li.rpmncs540-li-1.0.0.0-r632361.x86_64.rpm`

- 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

```
Router# configure
Router(config)# lawful-intercept disable
```

The `lawful-intercept disable` command is available on the router, only after installing and activating the `ncs5500-li.rpmncs540-li-1.0.0.0-r632361.x86_64.rpm`.

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.

Example: Configuring the Inband Management Plane Protection Feature

This example illustrates how to enable the MPP feature, which is disabled by default, for the purpose of lawful intercept.

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.

```
router# configure
router(config)# control-plane
router(config-cplt)# management-plane
router(config-mpp)# inband
router(config-mpp-inband)# interface loopback0
router(config-mpp-inband-Loopback0)# allow snmp
router(config-mpp-inband-Loopback0)# commit
router(config-mpp-inband-Loopback0)# exit
```
Enabling the Lawful Intercept SNMP Server Configuration

The following SNMP server configuration tasks enable the Cisco LI feature on a router running Cisco IOS XR Software by allowing the MD to intercept data sessions.

Configuration

```bash
router(config)# snmp-server engineID local 00:00:00:09:00:00:00:a1:61:6c:20:56
router(config)# snmp-server host 1.75.55.1 traps version 3 priv user-name udp-port 4444
router(config)# snmp-server user user-name li-group v3 auth md5 clear lab priv des56 clear lab
router(config)# snmp-server view li-view ciscoTap2MIB included
router(config)# snmp-server view li-view ciscoIpTapMIB included
router(config)# snmp-server view li-view snmp included
router(config)# snmp-server view li-view ifMIB included
router(config)# snmp-server view li-view 1.3.6.1.6.3.1.1.4.1 included
router(config)# snmp-server group li-group v3 auth read li-view write li-view notify li-view
```

**Note**
SNMP configuration must be removed while deactivating the LI RPM.

Additional Information on Lawful Intercept

**Interception Mode**

The lawful intercept operates in the **Global LI** mode.

In this mode, the taps are installed on all the line cards in the ingress direction. The lawful intercept is available on line cards where QoS peering is enabled. 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.

**Data Interception**

Data are intercepted in this manner:

- 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 IPv4 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.

Scale or Performance Values
The router supports the following scalability and performance values for lawful intercept:
• IPv4, or IPv6, or a combination of IPv4 and IPv6, lawful intercept tap limit is up to a maximum of 500 taps.
• The scale decreases, if port ranges are used in the taps.
• The IPv6 entries consume double the memory of the IPv4 entries. Hence, the IPv6 scale will reduce to half of the IPv4 scale.
• A maximum of 250 IPv4 MDs are supported.
• Interception rate is 1 Gbps best effort per Linecard NPU.

Intercepting IPv4 and IPv6 Packets
This section provides details for intercepting IPv4 and IPv6 packets supported on the 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
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 IPv4 UDP encapsulation. The replicated packets are forwarded to MD using UDP as the content delivery protocol.

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. The router does not support forwarding the same replicated packets to multiple MDs.

Note

Encapsulation types, such as RTP and RTP-NOR, are not supported.

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 and CISCO-IP-TAP-MIB to synchronize database view across RP and MD.

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.
Restrictions for IEEE 802.1X Port-Based Authentication

The following restrictions are applicable for IEEE 802.1X port-based authentication:

- 802.1X multi-host and 802.1X multi-auth are not supported.
- 802.1X VLAN assignment is not supported.
- Walled-garden VLAN and policies on authentication failures are not supported.
- Subinterfaces and VLAN-tagged traffic are not supported on the ports on which 802.1X port-based authentication is configured.

IEEE 802.1X Device Roles

The devices in the network have the following specific roles with IEEE 802.1X authentication:

- **Authenticator** - An entity that facilitates authentication of other entities attached to the same LAN.
- **Supplicant** - An entity at one end of a point-to-point LAN segment that seeks to be authenticated by an Authenticator attached to the other end of that link.
- **Authentication Server** - An entity that provides an authentication service to an Authenticator. Based on the credentials provided by the Supplicant, the server determines whether the Supplicant is authorized to access the services provided by the system in which the Authenticator resides.

Understanding 802.1X Port-Based Authentication

IEEE 802.1X port-based authentication is configured on Cisco NCS 540 Series Router to prevent unauthorized routers (supplicants) from gaining access to the network. An authentication server validates the supplicant
that is connected to an authenticator port, before the services offered by the client or the network is made available to the supplicant.

Until the supplicant is authenticated, the port is in *Unauthorized* state, and 802.1X access control allows only Extensible Authentication Protocol over LAN (EAPoL) packets through the port. EAPoL frames can have either default EtherType of 0x888E or Cisco-defined EtherType of 0x876F. After successful authentication of the supplicant, the port transitions to *Authorized* state, and normal traffic passes through the port for the authenticated client.

Periodic reauthentication can be enabled to use either the port-configured value or from authentication server. The authentication server communicates the reauthentication-timer value in Session-Timeout attribute, with the final RADIUS Access-Accept message. On 802.1X reauthentication failure, the port is blocked and moved back to the *Unauthorized* state.

If the link state of a port changes from up to down, or if an EAPOL-logoff frame is received, the port returns to the *Unauthorized* state.

The following figure shows the topology for IEEE 802.1X port-based authentication:

*Figure 2: Topology for IEEE 802.1X Port-Based Authentication*

---

**Configure 802.1X Port-Based Authentication**

To configure 802.1X port-based authentication, perform the following tasks:

1. Configure RADIUS server
2. Configure 802.1X authentication method
3. Create a 802.1X profile
4. Apply 802.1X profile on an interface
Configure RADIUS Server

To configure RADIUS server pre-shared keys, obtain the pre-shared key values for the remote RADIUS server and perform this task.

```
Router# configure terminal
Router(config)# radius-server host {ipv4-address|ipv6-address|host-name} key [0 Line 7]
   Key-value
Router(config)# radius-server vsa attribute ignore unknown
Router(config)# commit
```

Running Configuration

```
Router# show run radius
radius-server host 209.165.200.225 auth-port 1646
   key 7 094F471A1A0A57
radius-server vsa attribute ignore unknown
```

Configure 802.1X Authentication Method

You can configure 802.1X authentication method using RADIUS as the protocol. Only default AAA method is supported for 802.1X authentication.

```
Router# configure terminal
Router(config)# aaa authentication dot1x default group radius
Router(config)# commit
```

Running Configuration

```
Router# show run aaa
configure
   aaa authentication dot1x default group radius
```

Create 802.1X Profile

You can create multiple 802.1X profiles. The role of the node running 802.1X profile must be an authenticator.

```
Router# configure terminal
Router(config)# dot1x profile {name}
Router(config-dot1x-prof)# pae {authenticator}
Router(config-dot1x-prof)# authenticator timer reauth-time {reauth-time | server}
Router(config-dot1x-prof-auth)# commit
```

Running Configuration

```
Router# show run dot1x profile test_prof
dot1x profile test_prof
   pae authenticator
   authenticator
   timer reauth-time 3600
```

Apply 802.1X Profile on Interface

You can attach one of the 802.1X profiles on an interface.

```
Router# configure
Router(config)# interface interface-name
```
Verify 802.1X Port-Based Authentication

The 802.1X authentication can be verified using the following:

- Show command outputs
- Syslog messages

Show Command Outputs

The `show dot1x interface` command verifies whether the 802.1X port-based authentication is successful or not. If the authentication is successful, the traffic is allowed on the configured interface.

```
Router# show dot1x interface HundredGigE 0/0/1/0 detail
```

```
Dot1x info for HundredGigE 0/0/1/0
---------------------------------------------------------------
Interface short name : Hu 0/0/1/0
Interface handle : 0x4080
Interface MAC : 021a.9eeb.6a59
Ethertype : 888E
PAE : Authenticator

Dot1x Port Status : AUTHORIZED
Dot1x Profile : test_prof
L2 Transport : FALSE
Authenticator:
  Port Control : Enabled
  Config Dependency : Resolved
  Eap profile : None
  ReAuth : Disabled

Client List:
  Supplicant : 027e.15f2.cae7

Programming Status : Add Success
  Auth SM State : Authenticated

MKA Interface:
  Dot1x Tie Break Role : NA (Only applicable for PAE role both)
  EAP Based Macsec : Disabled
  MKA Start time : NA
  MKA Stop time : NA
  MKA Response time : NA
```
Syslog Messages

When 802.1x configuration is applied on an interface, the port becomes 802.1X controlled, and the following syslog message is displayed:

%L2-DOT1X-5-PORT_CONTROL_ENABLE_SUCCESS : Hu0/0/1/0 : Port Control Enabled

After successful authentication of supplicant, the following syslog messages are displayed:

%L2-DOT1X-5-AUTH_SUCCESS : Hu0/0/1/0 : Authentication successful for client 027E.15F2.CAE7

%L2-DOT1X-5-PORT_CONTROL_ADD_CLIENT_SUCCESS : Hu0/0/1/0 : Port Access Enabled For Client 027E.15F2.CAE7

When 802.1X port-based configuration is removed from an interface, the following syslog message is displayed:

%L2-DOT1X-5-PORT_CONTROL_DISABLE_SUCCESS : Hu0/0/1/0 : Port Control Disabled

When authentication fails, the following syslog messages are displayed:

%L2-DOT1X-5-AUTH_FAIL : Hu0/0/1/0 : Authentication fail for client 027E.15F2.CAE7

%L2-DOT1X-5-PORT_CONTROL_REMOVE_CLIENT_SUCCESS : Hu0/0/1/0 : Port Access Disabled For Client 027E.15F2.CAE7

When authentication server is unreachable, the following syslog message is displayed:

%L2-DOT1X-5-AAA_UNREACHABLE : Hu0/0/1/0 : AAA server unreachable for client 027E.15F2.CAE7, Retrying Authentication

When authentication method is not configured, the following syslog message is displayed:

%L2-DOT1X-4-NO_AUTHENTICATION_METHOD : Hu0/0/1/0 : No authentication method configured