# System Security Configuration Guide for Cisco CRS Routers, IOS XR Release 6.7.x

**First Published:** 2020-08-01

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

# CONTENTS

# Preface

**Note** This product has reached end-of-life status. For more information, see the End-of-Life and End-of-Sale Notices.

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

The preface contains the following sections:



## Changes to This Document

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

***Table 1: Changes to This Document***

| Date | Summary |
|---|---|
| August 2020 | Initial release of this document. |

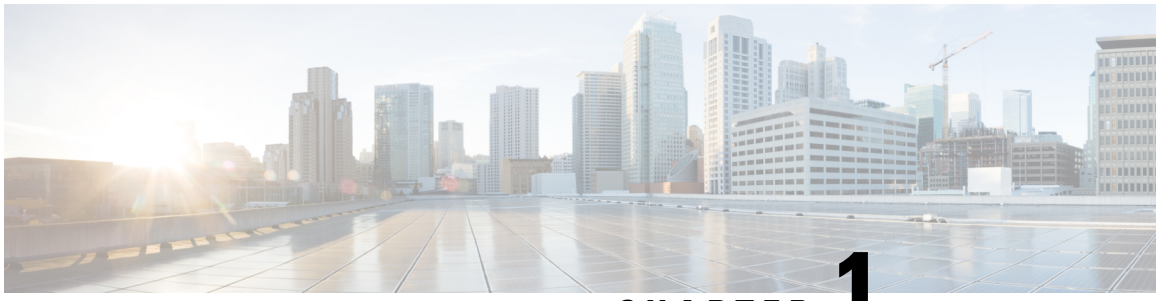
## Communications, Services, and Additional Information

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

- To find warranty information for a specific product or product family, access Cisco Warranty Finder.

### Cisco Bug Search Tool

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



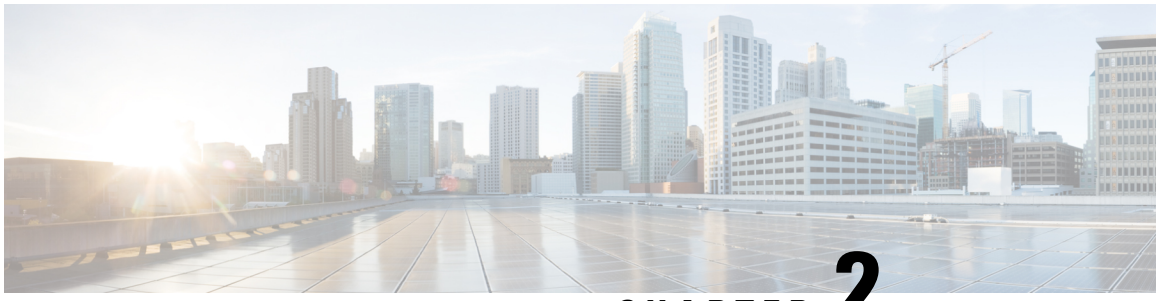
CHAPTER 1

# New and Changed Security Features



## System Security Features Added or Modified in IOS XR Release 6.7.x

| Feature | Description | Changed in Release | Where Documented |
|---|---|---|---|
| Automatic Generation of SSH Host-Key Pairs | This feature was introduced. | Release 6.7.2 | Automatic Generation of SSH Host-Key Pairs, on page 158 |



CHAPTER 2

# Configuring AAA Services

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

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

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

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

### Feature History for Configuring AAA Services

| Release | Modification |
|---|---|
| Release 3.2 | This feature was introduced. |

| Release | Modification |
|---|---|
| Release 3.3.0 | • Support for the RADIUS Dead-Server Detection feature was added.<br>• To enable interoperability based on Cisco IOS software, tasks must be marked as an optional attribute.<br>• Support was added on Cisco IOS XR to allow 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 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.<br>• A procedure on how to specify the task ID and user groups by using the CiscoSecure ACS was added.<br>• All references to owner secure domain router (SDR) were replaced with root SDR.<br>• Support was added to prompt the next logged-in user for a new username and password if all the users were deleted.<br>• The predefined task group *serviceadmin* was added.<br>• An example was added for RADIUS Vendor -Specific Attribute (VSA).<br>• EXEC authorization was added to the Administrative Access, on page 8 section. |
| Release 3.4.0 | • The **server-private** command was added to configure RADIUS server groups.<br>• Support for the Per VRF AAA feature was added.<br>• Support for generating interim accounting records was added. |
| Release 3.6.0 | The **eventmanager** keyword (fault manager) replaces the **exec** keyword to authorize event managers (fault managers) from the **aaa authorization** command. |
| Release 4.1.0 | Support for VRF aware TACACS+ was added. |
| Release 6.3.1 | Added the support for Type 8 and Type 9 passwords. |



# Information About Configuring AAA Services

This section lists all the conceptual information that a Cisco IOS XR software user must understand before configuring user groups and task groups through AAA or configuring Remote Authentication Dial-in User

Service (RADIUS) or TACACS+ servers. Conceptual information also describes what AAA is and why it is important.

# User, User Groups, and Task Groups

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

- User ID (ASCII string) that identifies the user uniquely across an administrative domain
- Length limitation of 253 characters for passwords and one-way encrypted secrets
- List of user groups (at least one) of which the user is a member (thereby enabling attributes such as task IDs). (See the Task IDs, on page 17 section)

## User Categories

Router users are classified into the following categories:

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

### Root System Users

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

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

- Configure secure domain routers.
- Create, delete, and modify root SDR users (after logging in to the secure domain router as the root system user). (See the Root SDR Users, on page 5 section.)
- Create, delete, and modify secure domain router users and set user task permissions (after logging in to the secure domain router as the root system user). (See the Secure Domain Router (SDR) Users, on page 6 section.)
- Access fabric racks or any router resource not allocated to a secure domain router, allowing the root system user to authenticate to any router node regardless of the secure domain router configurations.

### Root SDR Users

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

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

- Create, delete, and modify secure domain router users and their privileges for the SDR. (See the Secure Domain Router (SDR) Users, on page 6 section.)

- Create, delete, and modify user groups to allow access to the SDR.
- Manage nearly all aspects of the SDR.

A root SDR user cannot deny access to a root system user. (See the Root System Users, on page 5 section.)

### Secure Domain Router (SDR) Users

A SDR user has restricted access to an SDR as determined by the root-system user or root SDR user. The SDR user performs the day-to-day system and network management activities. The tasks that the secure domain router user is allowed to perform are determined by the task IDs associated with the user groups to which the SDR user belongs. (See the User Groups, on page 6 section.)

## User Groups

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

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

- A user group consists of the list of task groups that define the authorization for the users. All tasks, except cisco-support, are permitted by default for root system users. (See the Root System Users, on page 5 section.)
- Each user task can be assigned read, write, execute, or debug permission.

### Predefined User Groups

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

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

The user group root-system has root system users as the only members. (See the Root System Users, on page 5 section.) The root-system user group has predefined authorization; that is, it has the complete responsibility for root-system user-managed resources and certain responsibilities in other SDRs.

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

### User-Defined User Groups

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

### User Group Inheritance

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

## Task Groups

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

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

### Predefined Task Groups

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

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

### User-Defined Task Groups

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

### Group Inheritance

Task groups support inheritance from other task groups. (Similarly, a user group can derive attributes from another user group. See the User Groups, on page 6 section.) For example, when task group A inherits task group B, the new set of attributes of task group A is the union of A and B.

# Cisco IOS XR Software Administrative Model

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

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

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

## Administrative Access

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

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

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

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

To avoid a lockout, we recommend these:

- Before turning on TACACS+ command authorization or EXEC mode authorization on the console, make sure that the user who is configuring the authorization is logged in using the appropriate user permissions in the TACACS+ profile.
- If the security policy of the site permits it, use the **none** option for command authorization or EXEC mode authorization so that if the TACACS+ servers are not reachable, AAA rolls over to the **none** method, which permits the user to run the command.
- Make sure to allow local fallback when configuring AAA. See, Authorization Configuration, on page 51.
- If you prefer to commit the configuration on a trial basis for a specified time, you may do so by using the **commit confirmed** command, instead of direct **commit**.

# AAA Database

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

## Local Database

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

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

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

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

## Remote Database

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

# Remote AAA Configuration

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

## Client Configuration

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

## User Groups

User groups that are created in an external server are not related to the user group concept that is used in the context of local AAA database configuration on the router. The management of external TACACS+ server or RADIUS server user groups is independent, and the router does not recognize the user group structure. The remote user or group profiles may contain attributes that specify the groups (defined on the router) to which a user or users belong, as well as individual task IDs. For more information, see the Task IDs for TACACS+ and RADIUS Authenticated Users, on page 18 section.

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

### Task Groups

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

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

## AAA Configuration

This section provides information about AAA configuration.

### Method Lists

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

### Rollover Mechanism

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

The following methods are available:

- Local: Use the locally configured database (not applicable for accounting and certain types of authorization)
- TACACS+: Use a TACACS+ server (such as CiscoSecure ACS)
- RADIUS: Use a RADIUS server
- Line: Use a line password and user group (applicable only for authentication)
- None: Allow the request (not applicable for authentication)

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

### Server Grouping

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

## Authentication

Authentication is the most important security process by which a principal (a user or an application) obtains access to the system. The principal is identified by a username (or user ID) that is unique across an

administrative domain. The applications serving the user (such as or Management Agent) procure the username and the credentials from the user. AAA performs the authentication based on the username and credentials passed to it by the applications. The role of an authenticated user is determined by the group (or groups) to which the user belongs. (A user can be a member of one or more user groups.)

### Authentication of Root System User

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

### Authentication of Non-Owner Secure Domain Router User

When logging in from a non-owner secure domain router, the root system user must add the "@admin" suffix to the username. Using the "@admin" suffix sends the authentication request to the owner secure domain router for verification. The owner secure domain router uses the methods in the list-name **remote** for choosing the authentication method. The **remote** method list is configured using the **aaa authentication login remote** *method1 method2...* command. (See the Configuring AAA Method Lists, on page 48 section.)

### Authentication of Owner Secure Domain Router User

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

### Authentication of Secure Domain Router User

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

### Authentication Flow of Control

AAA performs authentication according to the following process:

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

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

Clients can obtain a user's permitted task IDs during authentication. This information is obtained by forming a union of all task group definitions specified in the user groups to which the user belongs. Clients using such information typically create a session for the user (such as an API session) in which the task ID set remains static. Both the EXEC mode and external API clients can use this feature to optimize their operations. EXEC

mode can avoid displaying the commands that are not applicable and an EMS application can, for example, disable graphical user interface (GUI) menus that are not applicable.

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

## Korn Shell Authentication

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

- For security reasons, ksh authentication allows only root-system users who have a secret configured. A root-system user with a normal password will not be authenticated because the normal password is two-way encrypted and poses a security risk because the password information is stored in the flash disk, which can be easily decrypted.
- Every time a root-system user with a secret is configured using the normal AAA CLI, that user is a valid ksh user and no separate configuration is required.
- Ksh does not authenticate TACACS+ or RADIUS users, even if they are root-system users.
- Ksh authentication uses a single user password database, which means when a root-system user on a dSC is configured using the normal AAA CLI, that user can log in using this username password in any card. This includes the RP, standby RP, LC, and SP.
- Ksh authentication cannot be turned off or bypassed after the card is booted. To bypass authentication, a user needs a reload of the card. (See the "Bypassing ksh Authentication" section for details).
- The ksh run from the console (using the **run** command) is not authenticated because the **run** command needs the root-system task ID. Because the user is already root-system, the user is not authenticated again.

### Bypassing ksh Authentication

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

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

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

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

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

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

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

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

### Authentication Failure

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

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

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

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

# Password Types

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

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

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

## Type 8 and Type 9 Passwords

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

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

# AAA Password Security for FIPS Compliance

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

### High Availability for AAA Password Security Policy

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

To configure AAA password policy, see Configure AAA Password Policy, on page 32.

## AAA Password Security Policies

AAA password security for FIPS compliance consists of these policies:

### Password Composition Policy

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

### Password Length Policy

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

### Password Lifetime Policy

The administrator can configure a maximum lifetime for the password, the value of which can be specified in years, months, days, hours, minutes and seconds. The configured password never expires if this parameter is not configured. The configuration remains intact even after a system reload. But, the password creation time is updated to the new time whenever the system reboots. For example, if a password is configured with a life time of one month, and if the system reboots on 29$^{th}$ day, then the password is valid for one more month after the system reboot. Once the configured lifetime expires, further action is taken based on the password expiry policy (see the section on Password Expiry Policy).

### Password Expiry Policy

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

- User is prompted to set the new password after successfully entering the expired password.

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

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

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

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

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

### Password Change Policy

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

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

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

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

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

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

### Service Provision after Authentication

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

### User Re-authentication Policy

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

### User Authentication Lockout Policy

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

This is a sample syslog when user is locked out:

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

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

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

### Password Policy Creation, Modification and Deletion

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

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

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

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

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

## Minimum Password Length for First User Creation

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

- When the Cisco Router is booted for the very first time.
- When the router is reloaded with no username configuration.

- When the already existing username configurations are deleted.

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

For more information about how to configure a first user, see Configure First User on Cisco Routers, on page 28.

# Task-Based Authorization

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

## Task IDs

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

Task IDs are assigned to users through the following means:

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

## General Usage Guidelines for Task IDs

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

***Table 2: Task ID Classes***

| Operation | Description |
|---|---|
| Read | Specifies a designation that permits only a read operation. |
| Write | Specifies a designation that permits a change operation and implicitly allows a read operation. |
| Execute | Specifies a designation that permits an access operation; for example ping and Telnet. |
| Debug | Specifies a designation that permits a debug operation. |

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

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

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

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

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

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

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

```
RP/0/RP0/CPU0:router# admin
RP/0/RP0/CPU0:router(admin)# show redundancy
```

# Task IDs for TACACS+ and RADIUS Authenticated Users

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

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

  See the "Task Maps, on page 18" section for more details.

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

  See the "Privilege Level Mapping, on page 20" section for more details.

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

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

## Task Maps

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

### Format of the Task String

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

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

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

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

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

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

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

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

## SUMMARY STEPS

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

## DETAILED STEPS

**Step 1** Enter your username and password.

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

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

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

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

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

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

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

**Example:**

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

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

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

**Example:**

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

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

**Example:**

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

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

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

**Example:**

```
Username:user2
Password:
RP/0/RP0/CPU0:router# show user tasks
No task ids available
```

## Privilege Level Mapping

For compatibility with TACACS+ daemons that do not support the concept of task IDs, AAA supports a mapping between privilege levels defined for the user in the external TACACS+ server configuration file and local user groups. Following TACACS+ authentication, the task map of the user group that has been mapped from the privilege level returned from the external TACACS+ server is assigned to the user. For example, if

a privilege level of 5 is returned from the external TACACS server, AAA attempts to get the task map of the local user group priv5. This mapping process is similar for other privilege levels from 1 to 13. For privilege level 15, the root-system user group is used; privilege level 14 maps to the user group owner-sdr.

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

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

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

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

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

# XML Schema for AAA Services

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

The following schema are published by AAA:

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

# About RADIUS

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

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

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

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

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

Use RADIUS in the following network environments that require access security:

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

## Network Security Situations in Which RADIUS is Unsuitable

RADIUS is not suitable in the following network security situations:

- Multiprotocol access environments. RADIUS does not support the following protocols:
  - AppleTalk Remote Access (ARA)
  - NetBIOS Frame Control Protocol (NBFCP)
  - NetWare Asynchronous Services Interface (NASI)
  - X.25 PAD connections

- Router-to-router situations. RADIUS does not provide two-way authentication. RADIUS can be used to authenticate from one router to a router other than a Cisco router if that router requires RADIUS authentication.
- Networks using a variety of services. RADIUS generally binds a user to one service model.

## RADIUS Operation

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

1. The user is prompted for and enters a username and password.
2. The username and encrypted password are sent over the network to the RADIUS server.
3. The user receives one of the following responses from the RADIUS server:
   a. ACCEPT—The user is authenticated.
   a. REJECT—The user is not authenticated and is prompted to reenter the username and password, or access is denied.
   a. CHALLENGE—A challenge is issued by the RADIUS server. The challenge collects additional data from the user.
   a. CHANGE PASSWORD—A request is issued by the RADIUS server, asking the user to select a new password.

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

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

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

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

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

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

# How to Configure AAA Services

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

## Prerequisites for Configuring AAA Services

The following are the prerequisites to configure AAA services:

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

## Restrictions for Configuring AAA Services

This section lists the restrictions for configuring AAA services.

### Compatibility

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

### Interoperability

Router administrators can use the same AAA server software and database (for example, CiscoSecure ACS) for the router and any other Cisco equipment that does not currently run Cisco IOS XR software. To support interoperability between the router and external TACACS+ servers that do not support task IDs, see the "Task IDs for TACACS+ and RADIUS Authenticated Users, on page 18" section.

## Configuring Task Groups

Task-based authorization employs the concept of a *task ID* as its basic element. A task ID defines the permission to execute an operation for a given user. Each user is associated with a set of permitted router operation tasks identified by task IDs. Users are granted authority by being assigned to user groups that are in turn associated with task groups. Each task group is associated with one or more task IDs selected from the Cisco CRS-1 set of available task IDs. The first configuration task in setting up an authorization scheme to configure the task groups, followed by user groups, followed by individual users.

### Task Group Configuration

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

The **inherit taskgroup** command may be used to derive permissions from another group. Circular references are detected and rejected. It is not possible to inherit from the root-system and owner-sdr predefined groups.

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

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

### Before you begin

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

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

## SUMMARY STEPS

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

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **taskgroup** *taskgroup-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# taskgroup beta` | Creates a name for a particular task group and enters task group configuration submode.<br>• Specific task groups can be removed from the system by specifying the **no** form of the **taskgroup** command. |
| **Step 3** | **description** *string*<br>**Example:**<br>`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** | **inherit taskgroup***taskgroup-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-tg)# inherit taskgroup sysadmin` | (Optional) Derives permissions from another task group and assigns them to the task group named in Step 2.<br>• Circular references are detected and rejected.<br>• To explicitly define permissions for the task group named in Step 2, omit Step 4 and go to Step 5. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **task** {**read** \| **write** \| **execute** \| **debug**} *taskid-name*<br>**Example:**<br>`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.<br>• Assigns **read** permission for any CLI or API invocations associated with that task ID and performed by a member of the task group.<br>• Specific task IDs can be removed from a task group by specifying the **no** prefix for the **task** command. |
| **Step 6** | Repeat Step 5 for each task ID to be associated with the task group named in Step 2. | — |
| **Step 7** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

### What to do next

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

# Configuring User Groups

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

Use the **inherit usergroup** command to inherit permissions from other user groups. The user group is inherited by the parent group and forms a union of all task IDs specified in those groups. Circular inclusions are detected and rejected.

### Before you begin

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

## SUMMARY STEPS

1. **configure**
2. **usergroup** *usergroup-name*
3. **description** *string*
4. **inherit usergroup** *usergroup-name*
5. **taskgroup** *taskgroup-name*
6. Repeat Step Step 5, on page 27 for each task group to be associated with the user group named in Step Step 2, on page 27.
7. Use the **commit** or **end** command.

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **usergroup** *usergroup-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# usergroup beta` | Creates a name for a particular user group and enters user group configuration submode.<br>• Specific user groups can be removed from the system by specifying the **no** form of the **usergroup** command. |
| **Step 3** | **description** *string*<br>**Example:**<br>`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 Step 2, on page 27. |
| **Step 4** | **inherit usergroup** *usergroup-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ug)# inherit usergroup sales` | • Circular inclusions are detected and rejected. Permissions may not be inherited from predefined user groups, such as root-system and owner-sdr.<br>• To explicitly define permissions for the user group named in Step Step 2, on page 27, omit Step Step 4, on page 27 and go to Step 5, on page 27. |
| **Step 5** | **taskgroup** *taskgroup-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ug)# taskgroup beta` | Associates the user group named in Step Step 2, on page 30with the task group named in this step.<br>• The user group takes on the configuration attributes (task ID list and permissions) already defined for the entered task group. |
| **Step 6** | Repeat Step Step 5, on page 27 for each task group to be associated with the user group named in Step Step 2, on page 27. | — |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

### What to do next

After completing configuration of a full set of user groups, configure individual users as described in the Configuring Users, on page 29 section.

# Configure First User on Cisco Routers

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

**Step 1.** Establish a connection to the Console port.

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

```
Enter root-system username
```

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

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

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

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

```
Enter secret
```

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

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

```
Enter secret again
```

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

**Step 5.** Log in to the router.

Establishes your access rights for the router management session.

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

For more information on minimum password length, see .

### Example

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

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

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

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

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

# Configuring Users

Perform this task to configure a user.

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

## SUMMARY STEPS

1. **configure**
2. **username** *user-name*
3. Do one of the following:
   - **password** [**0** | **7**] *password*
   -
4. **group** *group-name*
5. Repeat Step 4, on page 30 for each user group to be associated with the user specified in Step 2, on page 30.
6. Use the **commit** or **end** command.

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **username** *user-name*<br>**Example:**<br>`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.<br>• The *user-name* argument can be only one word. Spaces and quotation marks are not allowed. |
| **Step 3** | Do one of the following:<br>• **password** [**0** \| **7**] *password*<br>•<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-un)# password 0 pwd1`<br>or<br>`RP/0/RP0/CPU0:router(config-un)# secret 0 sec1` | Specifies a password for the user named in Step 2, on page 30.<br>• Use the **secret** command to create a secure login password for the user names specified in Step 2, on page 30.<br>• Entering **0** following the **password** command specifies that an unencrypted (clear-text) password follows. Entering **7** following the **password** command specifies that an encrypted password follows.<br>•<br>• Type **0** is the default for the **password** and **secret** commands. |
| **Step 4** | **group** *group-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-un)# group sysadmin` | Assigns the user named in Step 2, on page 30 to a user group that has already been defined through the **usergroup** command.<br>• The user takes on all attributes of the user group, as defined by that user group's association to various task groups.<br>• Each user must be assigned to at least one user group. A user may belong to multiple user groups. |
| **Step 5** | Repeat Step 4, on page 30 for each user group to be associated with the user specified in Step 2, on page 30. | — |
| **Step 6** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session. |

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

### What to do next

After completing configuration of a full set of users, configure router to use the RADIUS server communication or TACACS+ servers (See the Configuring Router to RADIUS Server Communication, on page 33 or Configuring a TACACS+ Server, on page 40 section.)

# Configure Type 8 and Type 9 Passwords

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

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

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

For more information about configuring Type 8 and Type 9 encryption methods, see Configuring Users, on page 29 section.

### Configuration Example

Directly configuring a Type 8 encrypted password:

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

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

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

Directly configuring a Type 9 encrypted password:

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

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

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

### Related Topics

- Type 8 and Type 9 Passwords, on page 13

### Associated Commands

- secret
- username

# Configure AAA Password Policy

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

### Configuration Example

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

```
RP/0/RP0/CPU0:router(config)#aaa password-policy test-policy
RP/0/RP0/CPU0:router(config-aaa)#min-length 8
RP/0/RP0/CPU0:router(config-aaa)#max-length 15
RP/0/RP0/CPU0:router(config-aaa)#lifetime months 3
RP/0/RP0/CPU0:router(config-aaa)#min-char-change 5
RP/0/RP0/CPU0:router(config-aaa)#authen-max-attempts 3
RP/0/RP0/CPU0:router(config-aaa)#lockout-time days 1
RP/0/RP0/CPU0:router(config-aaa)#commit

RP/0/RP0/CPU0:router(config)#username user1 password-policy test-policy password 0 pwd1
```

### Running Configuration

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

### Verification

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

```
RP/0/RP0/CPU0:router#show aaa password-policy

Fri Feb  3 16:50:58.086 EDT
Password Policy Name : test-policy
  Number of Users : 1
  Minimum Length : 8
  Maximum Length : 15
  Special Character Len : 0
  Uppercase Character Len : 0
  Lowercase Character Len : 1
  Numeric Character Len : 0
  Policy Life Time :
    seconds : 0
```

```
    minutes : 0
    hours : 0
    days : 0
    months : 3
    years : 0
  Lockout Time :
    seconds : 0
    minutes : 0
    hours : 0
    days : 1
    months : 0
    years : 0
  Character Change Len : 5
  Maximum Failure Attempts : 3
```

### Related Topic

- AAA Password Security for FIPS Compliance, on page 14

### Associated Commands

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

# Configuring Router to RADIUS Server Communication

This task configures router to RADIUS server communication.

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

- Hostname or IP address
- Authentication destination port
- Accounting destination port
- Retransmission value
- Timeout period
- Key string

RADIUS security servers are identified on the basis of their hostname or IP address, hostname and specific User Datagram Protocol (UDP) port numbers, or IP address and specific UDP port numbers. The combination of the IP address and UDP port numbers creates a unique identifier, allowing different ports to be individually defined as RADIUS hosts providing a specific AAA service. In other words, this unique identifier enables RADIUS requests to be sent to multiple UDP ports on a server at the same IP address. If two different host entries on the same RADIUS server are configured for the same service—for example, accounting—the second host entry configured acts as an automatic switchover backup to the first one. Using this example, if the first host entry fails to provide accounting services, the network access server tries the second host entry configured on the same device for accounting services. (The RADIUS host entries are tried in the order they are configured.)

A RADIUS server and a Cisco router use a shared secret text string to encrypt passwords and exchange responses.To configure RADIUS to use the AAA security commands, you must specify the host running the RADIUS server daemon and a secret text (key) string that it shares with the router.

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

You can configure a maximum of 30 global RADIUS servers.

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

## SUMMARY STEPS

1. **configure**
2. **radius-server host** {*hostname* | *ip address*} [**auth-port** *port-number*] [**acct-port** *port-number*] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*]
3. **radius-server retransmit** *retries*
4. **radius-server timeout** *seconds*
5. **radius-server key** {**0** *clear-text-key* | **7** *encrypted-key* | *clear-text-key*}
6. **radius source-interface** *type instance* [**vrf** *vrf-id*]
7. Repeat Step 2, on page 34 through Step 6, on page 35 for each external server to be configured.
8. Use the **commit** or **end** command.
9. show radius

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **radius-server host** {*hostname* \| *ip address*} [**auth-port** *port-number*] [**acct-port** *port-number*] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*]<br>**Example:**<br>Specifying a radius server hostname<br>`RP/0/RP0/CPU0:router(config)# radius-server host host1` | Specifies the hostname or IP address of the remote RADIUS server host. .<br>• Use the **auth-port** *port-number* option to configure a specific UDP port on this RADIUS server to be used solely for authentication.<br>• Use the **acct-port** *port-number* option to configure a specific UDP port on this RADIUS server to be used solely for accounting. |

| | Command or Action | Purpose |
|---|---|---|
| | | • 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.<br>• 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.<br>**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*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# radius-server retransmit 5` | Specifies the number of times the Cisco IOS XR software searches the list of RADIUS server hosts before giving up.<br>• In the example, the number of retransmission attempts is set to 5. |
| **Step 4** | **radius-server timeout** *seconds*<br>**Example:**<br>`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.<br>• 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*}<br>**Example:**<br>`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*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# radius source-interface POS 0/3/0/1` | (Optional) Forces RADIUS to use the IP address of a specified interface or subinterface for all outgoing RADIUS packets.<br>• 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. |

| | Command or Action | Purpose |
|---|---|---|
| | | The **vrf** keyword enables the specification on a per-VRF basis. |
| **Step 7** | Repeat Step 2, on page 34 through Step 6, on page 35 for each external server to be configured. | — |
| **Step 8** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 9** | show radius<br>**Example:**<br>`RP/0/RP0/CPU0:router# show radius` | (Optional) Displays information about the RADIUS servers that are configured in the system. |

### What to do next

After configuring router to RADIUS server communication, configure RADIUS server groups. (See the Configuring RADIUS Server Groups, on page 43 section.)

# Configuring RADIUS Dead-Server Detection

This task configures the RADIUS Dead-Server Detection feature.

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

You can configure the minimum amount of time, in seconds, that must elapse from the time that the router last received a valid packet from the RADIUS server to the time the server is marked as dead. If a packet has not been received since the router booted, and there is a timeout, the time criterion is treated as though it was met.

In addition, you can configure the number of consecutive timeouts that must occur on the router before the RADIUS server is marked as dead. If the server performs both authentication and accounting, both types of packets are included in the number. Improperly constructed packets are counted as though they are timeouts. Only retransmissions are counted, not the initial transmission. For example, each timeout causes one retransmission to be sent.

**Note** Both the time criterion and the tries criterion must be met for the server to be marked as dead.

The **radius-server deadtime** command specifies the time, in minutes, for which a server is marked as dead, remains dead, and, after this period, is marked alive even when no responses were received from it. When the dead criteria are configured, the servers are not monitored unless the **radius-server deadtime** command is configured

## SUMMARY STEPS

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

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **radius-server deadtime** *minutes*<br>**Example:**<br>`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. |
| **Step 3** | **radius-server dead-criteria time** *seconds*<br>**Example:**<br>`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. |
| **Step 4** | **radius-server dead-criteria tries** *tries*<br>**Example:**<br>`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. |
| **Step 5** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions: |

| | Command or Action | Purpose |
|---|---|---|
| | | • **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 6** | **show radius dead-criteria host** *ip address in IPv4 or IPv6 format* [**auth-port** *auth-port*] [**acct-port** *acct-port*]<br>**Example:**<br>`RP/0/RP0/CPU0:router# show radius dead-criteria host 172.19.192.80` | (Optional) Displays dead-server-detection information that has been requested for a RADIUS server at the specified IP address. |

# Configuring Per VRF AAA

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

## New Vendor-Specific Attributes (VSAs)

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

The Cisco IOS XR software RADIUS implementation supports one vendor-specific option using the format recommended in the specification. Cisco's vendor-ID is 9, and the supported option has vendor-type 1, which is named “cisco-avpair ” The value is a string of the following format:

```
protocol : attribute sep value *
```

“Protocol” is a value of the Cisco “protocol ” attribute for a particular type of authorization. “Attribute” and “value” are an appropriate attribute-value (AV) pair defined in the Cisco RADIUS specification, and “sep” is “=” for mandatory attributes and “*” for optional attributes.

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

***Table 3: Supported VSAs for Per VRF AAA***

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

| VSA Name | Value Type | Description |
|---|---|---|
| rad-serv | string | Indicates the IP address in IPv4 or IPv6 format, key, timeout, and retransmit number of a server and the group of the server.<br>The VSA syntax follows:<br>`rad-serv=a.b.c.d [key SomeKey] [auth-port X] [acct-port Y]`<br>`[retransmit V] [timeout W].`<br>Other than the IP address, all parameters are optional and are issued in any order. If the optional parameters are not specified, their default values are used.<br>The key cannot contain any spaces; for "retransmit V," "V" can range from 1 to 100; for "timeout W," the "W" can range from 1 to 1000. |
| rad-serv-vrf | string | Specifies the name of the VRF that is used to transmit RADIUS packets. The VRF name matches the name that was specified through the **vrf** command. |

This task configures RADIUS server groups per VRF. For information about configuring TACACS+ server groups per VRF, refer Configuring TACACS+ Server Groups, on page 45.

## SUMMARY STEPS

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

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **aaa group server radius** *group-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# aaa group server radius radgroup1`<br>`RP/0/RP0/CPU0:router(config-sg-radius)#` | Groups different server hosts into distinct lists and enters the server group configuration mode. |
| **Step 3** | **server-private** {*hostname* \| *ip-address in IPv4 or IPv6 format*} [**auth-port** *port-number*] [**acct-port** *port-number*] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*] | Configures the IP address of the private RADIUS server for the group. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>IP address in IPv4 format<br>`RP/0/RP0/CPU0:router(config-sg-radius)# server-private 10.1.1.1 timeout 5`<br>`RP/0/RP0/CPU0:router(config-sg-radius)# server-private 10.2.2.2 retransmit 3`<br>**Example:**<br>IP address in IPv6 format<br>`RP/0/RP0/CPU0:router(config-sg-radius)# server-private 2001:db8:a0b:12f0::1/64 timeout 5`<br>`RP/0/RP0/CPU0:router(config-sg-radius)# server-private 10.2.2.2 retransmit 3` | If private server parameters are not specified, global configurations are used. If global configurations are not specified, default values are used.<br>Both **auth-port** and **acct-port** keywords enter RADIUS server-group private configuration mode.<br>You can configure a maximum of 30 private servers per RADIUS server group. |
| **Step 4** | **vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-sg-radius)# vrf v2.44.com` | Configures the VRF reference of an AAA RADIUS server group.<br>**Note** Private server IP addresses can overlap with those configured globally and the VRF definitions can help to distinguish them. |
| **Step 5** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Configuring a TACACS+ Server

This task configures a TACACS+ server.

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

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

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

### SUMMARY STEPS

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

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **tacacs-server host** *host-name* **port** *port-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# tacacs-server host 209.165.200.226 port 51`<br>`RP/0/RP0/CPU0:router(config-tacacs-host)#` | Specifies a TACACS+ host server and optionally specifies a server port number.<br>• This option overrides the default, port 49. Valid port numbers range from 1 to 65535. |
| **Step 3** | **tacacs-server host** *host-name* **timeout** *seconds*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-tacacs-host)# tacacs-server host 209.165.200.226 timeout 30`<br>`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.<br>• 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*<br>**Example:**<br>`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.<br>• 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.<br>• (Optional) Entering **0** indicates that an unencrypted (clear-text) key follows. |

| | Command or Action | Purpose |
|---|---|---|
| | | • (Optional) Entering **7** indicates that an encrypted key follows.<br>• 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**<br>**Example:**<br>`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-server host** *host-name* **single-connection-idle-timeout** *timeout-in-seconds*<br>**Example:**<br>`RP/0/0RP0RSP0/CPU0:router:hostname(config)# tacacs-server host 209.165.200.226 single-connection-idle-timeout 60` | Sets the idle timeout value, in seconds, for the single TCP connection (that is created by configuring the **single-connection** command) to the TACACS+ server.<br>The range is 500 to 7200. |
| **Step 7** | **tacacs source-interface** *type instance* **vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# tacacs source-interface POS 0/4/0/0 vrf abc` | (Optional) Specifies the source IP address of a selected interface for all outgoing TACACS+ packets.<br>• 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.<br>• The **vrf** option specifies the Virtual Private Network (VPN) routing and forwarding (VRF) reference of an AAA TACACS+ server group. |
| **Step 8** | Repeat step 2 through step 6 for each external server to be configured. | — |
| **Step 9** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 10** | **show tacacs**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show tacacs` | (Optional) Displays information about the TACACS+ servers that are configured in the system. |

### What to do next

After configuring TACACS+ servers, configure TACACS+ server groups. (See the Configuring TACACS+ Server Groups, on page 45 section.)

# Configuring RADIUS Server Groups

This task configures RADIUS server groups.

The user can enter one or more **server** commands. The **server** command specifies the hostname or IP address of an external RADIUS server along with port numbers. When configured, this server group can be referenced from the AAA method lists (used while configuring authentication, authorization, or accounting). (See the Method Lists, on page 10 section.)

You can configure a maximum of:

- 30 servers per RADIUS server group
- 30 private servers per RADIUS server group

### Before you begin

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

## SUMMARY STEPS

1. **configure**
2. **aaa group server radius** *group-name*
3. **server** {*hostname* | *ip address in IPv4 or IPv6 format*} [**auth-port** *port-number*] [**acct-port** *port-number*]
4. Repeat Step 4, on page 44 for every external server to be added to the server group named in Step 3, on page 44.
5. **deadtime** *minutes*
6. Use the **commit** or **end** command.
7. **show radius server-groups** [*group-name* [**detail**]]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **aaa group server radius** *group-name*<br>**Example:**<br>`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 in IPv4 or IPv6 format*} [**auth-port** *port-number*] [**acct-port** *port-number*]<br>**Example:**<br>IP address in IPv4 format<br>`RP/0/RP0/CPU0:router(config-sg-radius)# server 192.168.20.0`<br>**Example:**<br>IP address in IPv6 format<br>`RP/0/RP0/CPU0:router(config-sg-radius)# server 2001:db8:a0b:12f0::1/64` | Specifies the hostname or IP address of an external RADIUS server.<br>• After the server group is configured, it can be referenced from the AAA method lists (used while configuring authentication, authorization, or accounting). |
| **Step 4** | Repeat Step 4, on page 44 for every external server to be added to the server group named in Step 3, on page 44. | — |
| **Step 5** | **deadtime** *minutes*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-sg-radius)# deadtime 1` | Configures the deadtime value at the RADIUS server group level.<br>• 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.<br>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<br>**Note** You can configure the group-level deadtime after the group is created. |
| **Step 6** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 7** | **show radius server-groups** [*group-name* [**detail**]]<br>**Example:** | (Optional) Displays information about each RADIUS server group that is configured in the system. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router# show radius server-groups` | |

### What to do next

After configuring RADIUS server groups, define method lists by configuring authentication, authorization, and accounting. (See the Configuring AAA Method Lists, on page 48 section.)

# Configuring TACACS+ Server Groups

This task configures TACACS+ server groups.

You can enter one or more **server** commands. The **server** command specifies the hostname or IP address of an external TACACS+ server. Once configured, this server group can be referenced from the AAA method lists (used while configuring authentication, authorization, or accounting). (See the Method Lists, on page 10 section.)

### Before you begin

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

## SUMMARY STEPS

1. **configure**
2. **aaa group server tacacs+** *group-name*
3. **server** {*hostname* | *ip address in IPv4 or IPv6 format*}
4. Repeat Step 3, on page 46 for every external server to be added to the server group named in Step 2, on page 45.
5. **server-private** {*hostname* | *ip-address in IPv4 or IPv6 format*} [**port** *port-number*] [**timeout** *seconds*] [**key** *string*]
6. **vrf** *vrf-name*
7. Use the **commit** or **end** command.
8. **show tacacs server-groups**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **aaa group server tacacs+** *group-name*<br>**Example:**<br>`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. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **server** {*hostname* \| *ip address in IPv4 or IPv6 format*}<br>**Example:**<br>IP address in IPv4 format<br>`RP/0/RP0/CPU0:router(config-sg-tacacs+)# server 192.168.100.0` | Specifies the hostname or IP address of an external TACACS+ server.<br>• When configured, this group can be referenced from the AAA method lists (used while configuring authentication, authorization, or accounting). |
| **Step 4** | Repeat Step 3, on page 46 for every external server to be added to the server group named in Step 2, on page 45. | — |
| **Step 5** | **server-private** {*hostname* \| *ip-address in IPv4 or IPv6 format*} [**port** *port-number*] [**timeout** *seconds*] [**key** *string*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-sg-tacacs+)# server-private 10.1.1.1 key a_secret` | Configures the IP address of the private TACACS+ server for the group server.<br>**Note**<br>• You can configure a maximum of 10 TACACS+ servers per server group.<br>• You can configure a maximum of 10 private TACACS+ servers.<br>• If private server parameters are not specified, global configurations are used. If global configurations are not specified, default values are used. |
| **Step 6** | **vrf** *vrf-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-sg-tacacs+)# vrf abc` | Specifies the Virtual Private Network (VPN) routing and forwarding (VRF) reference of an AAA TACACS+ server group. |
| **Step 7** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 8** | **show tacacs server-groups**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show tacacs server-groups` | (Optional) Displays information about each TACACS+ server group that is configured in the system. |

### What to do next

After configuring TACACS+ server groups, define method lists used by configuring authentication, authorization, and accounting. (See the Configuring AAA Method Lists, on page 48 section.)

# Configure Per VRF TACACS+ Server Groups

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

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

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

### Prerequisites

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

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

### Configuration Example

```
Router#configure

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

Router(config)# aaa group server tacacs+ tacgroup1

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

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

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

### Running Configuration

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

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

### Verify Per VRF TACACS+ Server Groups

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

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

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

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

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

### Associated Commands

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

# Configuring AAA Method Lists

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

This section contains the following procedures:

## Configuring Authentication Method Lists

This task configures method lists for authentication.

## Authentication Configuration

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

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

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

## Creation of a Series of Authentication Methods

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

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

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

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

### Before you begin

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

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

## SUMMARY STEPS

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

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **aaa authentication** {**login** | **ppp**} {**default** | *list-name* | **remote**} *method-list*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# aaa authentication login default group tacacs+` | Creates a series of authentication methods, or a method list.<br>• Using the **login** keyword sets authentication for login. Using the **ppp** keyword sets authentication for Point-to-Point Protocol.<br>• Entering the **default** keyword causes the listed authentication methods that follow this keyword to be the default list of methods for authentication.<br>• Entering a *list-name* character string identifies the authentication method list.<br>• Entering the **remote** keyword causes the listed authentication methods that follow this keyword to be the default list of methods for administrative authentication on a remote non-owner SDR.<br>**Note** The remote keyword is available only on the admin plane.<br>• 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:<br>• **group tacacs+**—Use a server group or TACACS+ servers for authentication<br>• **group radius**—Use a server group or RADIUS servers for authentication<br>• **group** *named-group*—Use a named subset of TACACS+ or RADIUS servers for authentication<br>• **local**—Use a local username or password database for authentication<br>• **line**—Use line password or user group for authentication<br>• The example specifies the **default** method list to be used for authentication. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| | | **end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 4** | Repeat Step 1 through Step 3 for every authentication method list to be configured. | — |

### What to do next

After configuring authentication method lists, configure authorization method lists. (See the Configuring Authorization Method Lists, on page 51 section).

## Configuring Authorization Method Lists

This task configures method lists for authorization.

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

### Authorization Configuration

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

**Note** The software attempts authorization with the next listed method only when there is no response or an error response (not a failure) from the previous method. If authorization fails at any point in this cycle—meaning that the security server or local username database responds by denying the user services—the authorization process stops and no other authorization methods are attempted.

Method lists are specific to the type of authorization being requested. Four types of AAA authorization are supported:

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

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

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

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

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

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

- Accept the request as is.
- Refuse authorization.

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

## Creation of a Series of Authorization Methods

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

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

- **none—**The router does not request authorization information; authorization is not performed over this line or interface.
- **local—**Uses local database for authorization.
- **group tacacs+—**Uses the list of all configured TACACS+ servers for authorization.
- **group radius—**Uses the list of all configured RADIUS servers for authorization.
- **group** *group-name*—Uses a named subset of TACACS+ or RADIUS servers for authorization.

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

For example,

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

## SUMMARY STEPS

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

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **aaa authorization** {**commands** | **eventmanager** | **exec** | **network**} {**default** | *list-name*} {**none** | **local** | **group** {**tacacs+** | **radius** | *group-name*}}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# aaa authorization commands listname1 group tacacs+` | Creates a series of authorization methods, or a method list.<br>• The **commands** keyword configures authorization for all EXEC mode shell commands. Command authorization applies to the EXEC mode commands issued by a user. Command authorization attempts authorization for all EXEC modecommands.<br>• The **eventmanager** keyword applies an authorization method for authorizing an event manager (fault manager).<br>• The **exec** keyword configures authorization for an interactive (EXEC mode) session.<br>• The **network** keyword configures authorization for network services like PPP or IKE.<br>• The **default** keyword causes the listed authorization methods that follow this keyword to be the default list of methods for authorization.<br>• 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 |

| | Command or Action | Purpose |
|---|---|---|
| | | sequence. The listed method list types can be any one of the following:<br>• **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.<br>• **local**—Uses local database for authorization.<br>• **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.<br>• **group radius**—Uses the list of all configured RADIUS servers for authorization.<br>• **group** *group-name*—Uses a named server group, a subset of TACACS+ or RADIUS servers for authorization as defined by the **aaa group server tacacs+** or **aaa group server radius** command. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

### What to do next

After configuring authorization method lists, configure accounting method lists. (See the Configuring Accounting Method Lists, on page 54 section.)

## Configuring Accounting Method Lists

This task configures method lists for accounting.

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

### Accounting Configuration

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

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

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

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

### Creation of a Series of Accounting Methods

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

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

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

### SUMMARY STEPS

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

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | Do one of the following:<br>• **aaa accounting** {**commands** \| **exec** \| **network**} {**default** \| *list-name*} {**start-stop** \| **stop-only**}<br>• {none \| method}<br>**Example:**<br>`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.<br>Creates a series of accounting methods, or a method list.<br>• The **commands** keyword enables accounting for EXEC mode shell commands.<br>• The **exec** keyword enables accounting for an interactive (EXEC mode) session.<br>• The **network** keyword enables accounting for all network-related service requests, such as Point-to-Point Protocol (PPP).<br>• The **default** keyword causes the listed accounting methods that follow this keyword to be the default list of methods for accounting.<br>• A *list-name* character string identifies the accounting method list.<br>• 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.<br>• The **stop-only** keyword sends a "stop accounting" notice at the end of the requested user process.<br>• The **none** keyword states that no accounting is performed.<br>• 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:<br>  • **group tacacs+**—Use the list of all configured TACACS+ servers for accounting.<br>  • **group radius—**Use the list of all configured RADIUS servers for accounting.<br>  • **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.<br>• 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. |

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

### What to do next

After configuring method lists, apply those method lists. (See the Applying Method Lists for Applications, on page 58 section.)

## Generating Interim Accounting Records

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

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

### SUMMARY STEPS

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

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **aaa accounting update** {**newinfo** | **periodic** *minutes*}<br>**Example:** | Enables periodic interim accounting records to be sent to the accounting server.<br>• If the **newinfo** keyword is used, interim accounting records are sent to the accounting server every time |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config)# aaa accounting update periodic 30` | 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.<br>• 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.<br>**Caution** The **periodic** keyword causes heavy congestion when many users are logged in to the network. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Applying Method Lists for Applications

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

This section contains the following procedures:

## Enabling AAA Authorization

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

### Method List Application

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

### SUMMARY STEPS

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

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **line** {**aux** | **console** | **default** | **template** *template-name*}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# line console` | Enters line template configuration mode. |
| **Step 3** | **authorization** {**commands** | **exec**} {**default** | *list-name*}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-line)# authorization commands listname5` | Enables AAA authorization for a specific line or group of lines.<br>• The **commands** keyword enables authorization on the selected lines for all commands.<br>• The **exec** keyword enables authorization for an interactive (EXEC mode) session.<br>• Enter the **default** keyword to apply the name of the default method list, as defined with the **aaa authorization** command.<br>• Enter the name of a list of authorization methods to use. If no list name is specified, the system uses the default. The list is created with the **aaa authorization** command.<br>• The example enables command authorization using the method list named listname5. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

### What to do next

After applying authorization method lists by enabling AAA authorization, apply accounting method lists by enabling AAA accounting. (See the Enabling Accounting Services, on page 60 section.)

## Enabling Accounting Services

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

### SUMMARY STEPS

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

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **line { aux \| console \| default \| template template-name**}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# line console` | Enters line template configuration mode. |
| **Step 3** | **accounting** {**commands** \| **exec**} {**default** \| *list-name*}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-line)# accounting commands listname7` | Enables AAA accounting for a specific line or group of lines.<br>• The **commands** keyword enables accounting on the selected lines for all EXEC mode shell commands.<br>• The **exec** keyword enables accounting for an interactive (EXEC mode) session.<br>• Enter the **default** keyword to apply the name of the default method list, as defined with the **aaa accounting** command.<br>• 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.<br>• The example enables command accounting using the method list named listname7. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions: |

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

### What to do next

After applying accounting method lists by enabling AAA accounting services, configure login parameters. (See the Configuring Login Parameters, on page 61 section.)

# Configuring Login Parameters

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

### SUMMARY STEPS

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

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **line template** *template-name*<br>**Example:**<br>`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*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-line)# timeout login response 20` | Sets the interval that the server waits for reply to a login.<br>• The *seconds* argument specifies the timeout interval (in seconds) from 0 to 300. The default is 30 seconds.<br>• The example shows how to change the interval timer to 20 seconds. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| | | **end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Configuration Examples for Configuring AAA Services

This section provides the following configuration example:

## Configuring AAA Services: Example

The following examples show how to configure AAA services.

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

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

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

```
aaa authentication ppp default local
```

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

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

username user2
secret lab
exit
```

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

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

usergroup uga
taskgroup tga
description usergroup uga
exit
```

Username user2 is configured to inherit from user group uga.

```
username user2
group uga
exit
```

Three TACACS servers are configured.

```
tacacs-server host 1.1.1.1 port 1 key abc
tacacs-server host 2.2.2.2 port 2 key def
tacacs-server host 3.3.3.3 port 3 key ghi
```

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

```
usergroup priv5
taskgroup operator
exit
```

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

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

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

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

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

```
aaa default-taskgroup tga
```

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

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

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

```
aaa group server tacacs+ abc
server 3.3.3.3
exit
```

# Model-based AAA

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

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

## Prerequisites for Model Based AAA

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

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

## Initial Operation

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

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

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

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

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

```
</username>
</usernames>
</aaa>

<nacm xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-nacm-cfg">
<read-default>permit</read-default>
<write-default>permit</write-default>
<exec-default>permit</exec-default>
<enable-external-groups>true</enable-external-groups>
<groups>
    <group>
    <name>nacm_group</name>
    <user-name>lab</user-name>
    </group>
</groups>
<rule-list>
<name>Rule-list-1</name>
<group>Group_nacm_0_test</group>
<rule>
    <name>Rule-1</name>
    <access-operations>read</access-operations>
    <action>permit</action>
    <module-name>ietf-netconf-acm</module-name>
    <rpc-name>edit-config</rpc-name>
                <access-operations>*</access-operations>
                <path>/</path>
                <action>permit</action>
            </rule>
       </rule-list>
</nacm>
```

# NACM Configuration Management and Persistence

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

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

# Overview of Configuring NACM

Here are the steps involved in configuring NACM:

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

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

# NACM Rules

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

- Global Rules—It includes the following:
  - Enable/Disable NACM
  - Read-Default
  - Write-Default
  - Exec-Default
  - Enable External Groups
- Access Control Rules—It includes the following:
  - Module (used along with protocol rule / data node rule)
  - Protocol
  - Data Node

The following table lists the rules and access operations:

| Operation | Description |
|---|---|
| all | Rule is applied to all types of protocol operations |
| create | Rule is applied to all protocol operations, which create a new data node such as edit-config operation |
| read | Rule is applied to all protocol operations, which reads the data node such as get, get-config or notification |
| update | Rule is applied to all protocol operations, which alters a data node such as edit-config operation |
| exec | Rule is applied to all exec protocol access operations such as action RPC |
| delete | Rule is applied to all protocol operations that removes a data node |

**Note** Before enabling NACM using NETCONF RPC, any user with access to the system can create NACM groups and rules. However, after NACM is enabled, only authorised users can change the NACM configurations.

**Example: Configure Global Rules**

**YANG Data Model:** You must configure NACM groups and NACM rulelist before configuring NACM rules. The following sample configuration shows a NACM group configuration:

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" >
<edit-config>
  <target><candidate/></target>
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
    <groups>
      <group>
        <name>group1</name>
        <user-name>user1</user-name>
        <user-name>user2</user-name>
        <user-name>user3</user-name>
      </group>
    </groups>
  </nacm>
</config>
</edit-config>
</rpc>
```

The following sample configuration shows a NACM rule list configuration:

```
<rpc
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"message-id="101">
<edit-config>
  <target>
    <candidate/>
  </target>
 <config>
  <nacm xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-aaa-nacm-cfg">
  <rulelist-classes>
  <rulelist-class>
  <ordering-index>1</ordering-index>
 <rulelist-name>GlobalRule</rulelist-name>
 <group-names>
  <group-name>root-system</group-name>
   <group-name>AdminUser</group-name>
 </group-names>
 </rulelist-class>
 </rulelist-classes>
</nacm>
</config>
</edit-config>
</rpc>
```

### Example: Configure NACM Global Rules

### YANG Data Model:

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" >
<edit-config>
  <target><candidate/></target>
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
    <read-default>permit</read-default>
    <write-default>permit</write-default>
    <exec-default>permit</exec-default>
    <enable-external-groups>false</enable-external-groups>
  </nacm>
</config>
</edit-config>
</rpc>
```

### Example: Configure Access Control Rules

### YANG Data Model:

```
<rpc message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" >
<edit-config>
<target><candidate/></target>
<config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
    <rule-list>
     <name>GlobalRule</name>
     <rule>
       <name>rule1</name>
       <module-name>ietf-netconf-acm</module-name>
       <rpc-name>edit-config</rpc-name>
       <access-operations>*</access-operations>
       <action>permit</action>
     </rule>
    <rule>
      <name>rule2</name>
      <module-name>ietf-netconf-acm</module-name>
      <rpc-name>get-config</rpc-name>
      <access-operations>create read update exec</accessoperations>
      <action>permit</action>
    </rule>
   </rule-list>
  </nacm>
 </config>
 </edit-config>
</rpc>
```

**Note** '*' refers to all operations.

### Example: Configure NACM Data Node Rules

```
<rpc message-id="101"xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" >
<edit-config>
<target><candidate/></target>
  <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
    <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
     <rule-list>
        <name>GlobalRule</name>
      <rule>
       <name>rule4</name>
       <module-name>*</module-name>
       <path>/nacm/groups/group</path>
       <access-operations>*</access-operations>
       <action>permit</action>
      </rule>
      <rule>
       <name>rule5</name>
       <module-name>ietf-netconf-acm</module-name>
       <path>/nacm/rule-list</path>
       <access-operations>read</access-operations>
       <action>deny</action>
      </rule>
     </rule-list>
   </nacm>
  </config>
```

```
 </edit-config>
</rpc>
```

**Note** '*' refers to all modules, and all operations.

# Enabling NACM

NACM is disabled on the router by default. Users with root-lr or 'aaa' write task privilege users can enable/disable the NACM via CLI.

To enable NACM, use the following command in the Global configuration mode:

```
Router(config)#aaa authorization nacm default local
```

### Verification

Use the **show nacm summary** command to verify the default values after enabling NACM:

```
Router# show nacm summary
Mon Jan 15 16:47:43.549 UTC
 NACM SUMMARY
 ------------------------------------------------------------------------------
 Enable Nacm : True
 Enable External Groups : True
 Number of Groups : 0
 Number of Users : 0
 Number of Rules : 0
 Number of Rulelist : 0
 Default Read : permit
 Default Write : deny
 Default Exec : permit
 Denied Operations : 0
 Denied Data Writes : 0
 Denied Notifications : 0
```

### Associated Commands

- Router#**show nacm summary**
- Router#**show nacm users [user-name]**
- Router#**show nacm rule-list [rule-list-name] [rule [rule-name]]**
- Router#**show nacm groups [group-name]secret**

# Verify the NACM Configurations

Use the **show nacm summary** command to verify the NACM configurations:

```
Router# show nacm summary
Mon Jan 15 17:02:46.696 UTC
 NACM SUMMARY
 ------------------------------------------------------------------------------
 Enable Nacm : True
```

```
 Enable External Groups : True
 Number of Groups : 3
 Number of Users : 3
 Number of Rules : 4
 Number of Rulelist : 2
 Default Read : permit
 Default Write : permit
 Default Exec : permit
 Denied Operations : 1
 Denied Data Writes : 0
 Denied Notifications : 0
-------------------------------------------------------------------------------
```

### Associated Commands

- Router#**show nacm summary**
- Router#**show nacm users [user-name]**
- Router#**show nacm rule-list [rule-list-name] [rule [rule-name]]**
- Router#**show nacm groups [group-name]secret**

# Disabling NACM

There are two ways you can disable NACM. Use one of the following commands:

Configuring NACM authorization as none:

```
Router(config)# aaa authorization nacm default none
```

or

Using no form of AAA authorization command:

```
Router(config)# no aaa authorization nacm default
```

### Verification

Use the **show nacm summary** command to verify the default values after disabling NACM:

```
Router# show nacm summary

Mon Jan 15 17:02:46.696 UTC
 NACM SUMMARY
 -------------------------------------------------------------------------------
 Enable Nacm : False
 Enable External Groups : True
 Number of Groups : 0
 Number of Users : 0
 Number of Rules : 0
 Number of Rulelist : 0
 Default Read : permit
 Default Write : deny
 Default Exec : permit
 Denied Operations : 0
 Denied Data Writes : 0
 Denied Notifications : 0
```

# Additional References

The following sections provide references related to configuring AAA services.

### Related Documents

| Related Topic | Document Title |
|---|---|
| AAA services commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *Authentication, Authorization, and Accounting Commands on* the Cisco IOS XR Software in the *System Security Command Reference for Cisco CRS Routers* |

### Standards

| Standards | Title |
|---|---|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

### MIBs

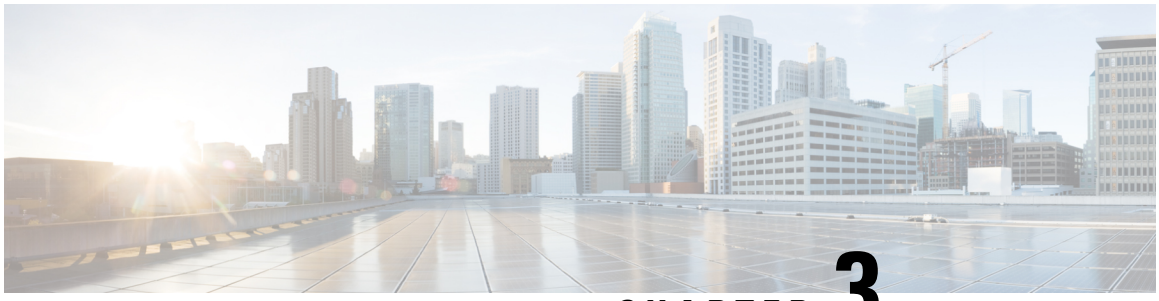
| MIBs | MIBs Link |
|---|---|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml |

### RFCs

| RFCs | Title |
|---|---|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 3

# Implementing Certification Authority Interoperability

Certification authority (CA) interoperability is provided in support of the IP Security (IPSec), Secure Socket Layer (SSL), and Secure Shell (SSH) protocols. CA interoperability permits Cisco CRS Router devices and CAs to communicate so that your device can obtain and use digital certificates from the CA. Although IPSec can be implemented in your network without the use of a CA, using a CA provides manageability and scalability for IPSec.

This module describes the tasks that you need to implement CA interoperability on your Cisco IOS XR software network.

**Note** For a complete description of the public key infrastructure (PKI) commands used in this chapter, refer to the *Public Key Infrastructure Commands* module in *System Security Command Reference for Cisco CRS Routers*.

### Feature History for Implementing Certification Authority Interoperability

| Release | Modification |
|---|---|
| Release 2.0 | This feature was introduced. |
| Release 3.4.0 | A procedure was added on how to declare the trustpoint certification authority (CA). |
| Release 5.2 | A section was added on trust pool management |

# 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 *System Management Configuration Guide for Cisco CRS Routers*
- You need to have a CA available to your network before you configure this interoperability feature. The CA must support Cisco Systems PKI protocol, the simple certificate enrollment protocol (SCEP) (formerly called certificate enrollment protocol [CEP]).

# Restrictions for Implementing Certification Authority

Cisco IOS XR software does not support CA server public keys greater than 2048 bits.

# Information About Implementing Certification Authority

To implement CA, you need to understand the following concepts:

## Supported Standards for Certification Authority Interoperability

Cisco supports the following standards:

- IPSec—IP Security Protocol. IPSec is a framework of open standards that provides data confidentiality, data integrity, and data authentication between participating peers. IPSec provides these security services at the IP layer; it uses Internet Key Exchange (IKE) to handle negotiation of protocols and algorithms based on local policy, and to generate the encryption and authentication keys to be used by IPSec. IPSec can be used to protect one or more data flows between a pair of hosts, a pair of security gateways, or a security gateway and a host.
- IKE—A hybrid protocol that implements Oakley and Skeme key exchanges inside the Internet Security Association Key Management Protocol (ISAKMP) framework. Although IKE can be used with other protocols, its initial implementation is with the IPSec protocol. IKE provides authentication of the IPSec peers, negotiates IPSec keys, and negotiates IPSec security associations (SAs).
- Public-Key Cryptography Standard #7 (PKCS #7)—A standard from RSA Data Security Inc. used to encrypt and sign certificate enrollment messages.
- Public-Key Cryptography Standard #10 (PKCS #10)—A standard syntax from RSA Data Security Inc. for certificate requests.
- RSA keys—RSA is the public key cryptographic system developed by Ron Rivest, Adi Shamir, and Leonard Adelman. RSA keys come in pairs: one public key and one private key.

- SSL—Secure Socket Layer protocol.
- X.509v3 certificates—Certificate support that allows the IPSec-protected network to scale by providing the equivalent of a digital ID card to each device. When two devices want to communicate, they exchange digital certificates to prove their identity (thus removing the need to manually exchange public keys with each peer or specify a shared key at each peer). These certificates are obtained from a CA. X.509 as part of the X.500 standard of the ITU.

# Certification Authorities

The following sections provide background information about CAs:

## Purpose of CAs

CAs are responsible for managing certificate requests and issuing certificates to participating IPSec network devices. These services provide centralized key management for the participating devices.

CAs simplify the administration of IPSec network devices. You can use a CA with a network containing multiple IPSec-compliant devices, such as routers.

Digital signatures, enabled by public key cryptography, provide a means of digitally authenticating devices and individual users. In public key cryptography, such as the RSA encryption system, each user has a key pair containing both a public and a private key. The keys act as complements, and anything encrypted with one of the keys can be decrypted with the other. In simple terms, a signature is formed when data is encrypted with a user's private key. The receiver verifies the signature by decrypting the message with the sender's public key. The fact that the message could be decrypted using the sender's public key indicates that the holder of the private key, the sender, must have created the message. This process relies on the receiver's having a copy of the sender's public key and knowing with a high degree of certainty that it does belong to the sender and not to someone pretending to be the sender.

Digital certificates provide the link. A digital certificate contains information to identify a user or device, such as the name, serial number, company, department, or IP address. It also contains a copy of the entity's public key. The certificate is itself signed by a CA, a third party that is explicitly trusted by the receiver to validate identities and to create digital certificates.

To validate the signature of the CA, the receiver must first know the CA's public key. Normally, this process is handled out-of-band or through an operation done at installation. For instance, most web browsers are configured with the public keys of several CAs by default. IKE, an essential component of IPSec, can use digital signatures to authenticate peer devices for scalability before setting up SAs.

Without digital signatures, a user must manually exchange either public keys or secrets between each pair of devices that use IPSec to protect communication between them. Without certificates, every new device added to the network requires a configuration change on every other device with which it communicates securely. With digital certificates, each device is enrolled with a CA. When two devices want to communicate, they exchange certificates and digitally sign data to authenticate each other. When a new device is added to the network, a user simply enrolls that device with a CA, and none of the other devices needs modification. When the new device attempts an IPSec connection, certificates are automatically exchanged and the device can be authenticated.

## IPSec Without CAs

Without a CA, if you want to enable IPSec services (such as encryption) between two Cisco routers, you must first ensure that each router has the key of the other router (such as an RSA public key or a shared key). This requirement means that you must manually perform one of the following operations:

- At each router, enter the RSA public key of the other router.
- At each router, specify a shared key to be used by both routers.

If you have multiple Cisco routers in a mesh topology and want to exchange IPSec traffic passing among all of those routers, you must first configure shared keys or RSA public keys among all of those routers.

Every time a new router is added to the IPSec network, you must configure keys between the new router and each of the existing routers.

Consequently, the more devices there are that require IPSec services, the more involved the key administration becomes. This approach does not scale well for larger, more complex encrypting networks.

## IPSec with CAs

With a CA, you need not configure keys between all the encrypting routers. Instead, you individually enroll each participating router with the CA, requesting a certificate for the router. When this enrollment has been accomplished, each participating router can dynamically authenticate all the other participating routers.

To add a new IPSec router to the network, you need only configure that new router to request a certificate from the CA, instead of making multiple key configurations with all the other existing IPSec routers.

## IPSec with Multiple Trustpoint CAs

With multiple trustpoint CAs, you no longer have to enroll a router with the CA that issued a certificate to a peer. Instead, you configure a router with multiple CAs that it trusts. Thus, a router can use a configured CA (a trusted root) to verify certificates offered by a peer that were not issued by the same CA defined in the identity of the router.

Configuring multiple CAs allows two or more routers enrolled under different domains (different CAs) to verify the identity of each other when using IKE to set up IPSec tunnels.

Through SCEP, each router is configured with a CA (the enrollment CA). The CA issues a certificate to the router that is signed with the private key of the CA. To verify the certificates of peers in the same domain, the router is also configured with the root certificate of the enrollment CA.

To verify the certificate of a peer from a different domain, the root certificate of the enrollment CA in the domain of the peer must be configured securely in the router.

During IKE phase one signature verification, the initiator will send the responder a list of its CA certificates. The responder should send the certificate issued by one of the CAs in the list. If the certificate is verified, the router saves the public key contained in the certificate on its public key ring.

With multiple root CAs, Virtual Private Network (VPN) users can establish trust in one domain and easily and securely distribute it to other domains. Thus, the required private communication channel between entities authenticated under different domains can occur.

## How IPSec Devices Use CA Certificates

When two IPSec routers want to exchange IPSec-protected traffic passing between them, they must first authenticate each other—otherwise, IPSec protection cannot occur. The authentication is done with IKE.

*Without* a CA, a router authenticates itself to the remote router using either RSA-encrypted nonces or preshared keys. Both methods require keys to have been previously configured between the two routers.

*With* a CA, a router authenticates itself to the remote router by sending a certificate to the remote router and performing some public key cryptography. Each router must send its own unique certificate that was issued and validated by the CA. This process works because the certificate of each router encapsulates the public key of the router, each certificate is authenticated by the CA, and all participating routers recognize the CA as an authenticating authority. This scheme is called IKE with an RSA signature.

Your router can continue sending its own certificate for multiple IPSec sessions and to multiple IPSec peers until the certificate expires. When its certificate expires, the router administrator must obtain a new one from the CA.

When your router receives a certificate from a peer from another domain (with a different CA), the certificate revocation list (CRL) downloaded from the CA of the router does not include certificate information about the peer. Therefore, you should check the CRL published by the configured trustpoint with the Lightweight Directory Access Protocol (LDAP) URL to ensure that the certificate of the peer has not been revoked.

To query the CRL published by the configured trustpoint with the LDAP URL, use the **query url** command in trustpoint configuration mode.

### CA Registration Authorities

Some CAs have a registration authority (RA) as part of their implementation. An RA is essentially a server that acts as a proxy for the CA so that CA functions can continue when the CA is offline.

# How to Implement CA Interoperability

This section contains the following procedures:

## Configuring a Router Hostname and IP Domain Name

This task configures a router hostname and IP domain name.

You must configure the hostname and IP domain name of the router if they have not already been configured. The hostname and IP domain name are required because the router assigns a fully qualified domain name (FQDN) to the keys and certificates used by IPSec, and the FQDN is based on the hostname and IP domain name you assign to the router. For example, a certificate named router20.example.com is based on a router hostname of router20 and a router IP domain name of example.com.

#### SUMMARY STEPS

1. **configure**
2. **hostname** *name*
3. **domain name domain-name**
4. Use the **commit** or **end** command.

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **hostname** *name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# hostname myhost` | Configures the hostname of the router. |
| **Step 3** | **domain name domain-name**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# domain name mydomain.com` | Configures the IP domain name of the router. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Generating an 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.

**SUMMARY STEPS**

1. **crypto key generate rsa** [**usage keys** | **general-keys**] [**keypair-label**]
2. **crypto key zeroize rsa** [**keypair-label**]
3. show crypto key mypubkey rsa

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **crypto key generate rsa** [**usage keys** \| **general-keys**] [**keypair-label**]<br>**Example:**<br>`RP/0/RP0/CPU0:router# crypto key generate rsa general-keys` | Generates RSA key pairs.<br>• Use the **usage keys** keyword to specify special usage keys; use the **general-keys** keyword to specify general-purpose RSA keys.<br>• The *keypair-label* argument is the RSA key pair label that names the RSA key pairs. |
| **Step 2** | **crypto key zeroize rsa** [**keypair-label**]<br>**Example:**<br>`RP/0/RP0/CPU0:router# crypto key zeroize rsa key1` | (Optional) Deletes all RSAs from the router.<br>• Under certain circumstances, you may want to delete all RSA keys from you router. For example, if you believe the RSA keys were compromised in some way and should no longer be used, you should delete the keys.<br>• To remove a specific RSA key pair, use the *keypair-label* argument. |
| **Step 3** | show crypto key mypubkey rsa<br>**Example:**<br>`RP/0/RP0/CPU0:router# show crypto key mypubkey rsa` | (Optional) Displays the RSA public keys for your router. |

# Importing a Public Key to the Router

This task imports a public key to the router.

A public key is imported to the router to authenticate the user.

**SUMMARY STEPS**

1. **crypto key import authentication rsa** [**usage keys** | **general-keys**] [**keypair-label**]
2. show crypto key mypubkey rsa

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **crypto key import authentication rsa** [**usage keys** \| **general-keys**] [**keypair-label**]<br>**Example:**<br>`RP/0/RP0/CPU0:router# crypto key import authentication rsa general-keys` | Generates RSA key pairs.<br>• Use the **usage keys** keyword to specify special usage keys; use the **general-keys** keyword to specify general-purpose RSA keys.<br>• The *keypair-label* argument is the RSA key pair label that names the RSA key pairs. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | show crypto key mypubkey rsa<br>**Example:**<br>`RP/0/RP0/CPU0:router# show crypto key mypubkey rsa` | (Optional) Displays the RSA public keys for your router. |

# Declaring a Certification Authority and Configuring a Trusted Point

This task declares a CA and configures a trusted point.

### SUMMARY STEPS

1. **configure**
2. **crypto ca trustpoint ca-name**
3. **enrollment url CA-URL**
4. **query url LDAP-URL**
5. **enrollment retry period minutes**
6. **enrollment retry count number**
7. **rsakeypair keypair-label**
8. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **crypto ca trustpoint ca-name**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# crypto ca trustpoint myca` | Declares a CA.<br>• Configures a trusted point with a selected name so that your router can verify certificates issued to peers.<br>• Enters trustpoint configuration mode. |
| **Step 3** | **enrollment url CA-URL**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-trustp)# enrollment url http://ca.domain.com/certsrv/mscep/mscep.dll` | Specifies the URL of the CA.<br>• The URL should include any nonstandard cgi-bin script location. |
| **Step 4** | **query url LDAP-URL**<br>**Example:**<br>`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. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **enrollment retry period minutes**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-trustp)# enrollment retry period 2` | (Optional) Specifies a retry period.<br>• 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.<br>• Range is from 1 to 60 minutes. Default is 1 minute. |
| **Step 6** | **enrollment retry count number**<br>**Example:**<br>`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.<br>• The range is from 1 to 100. |
| **Step 7** | **rsakeypair keypair-label**<br>**Example:**<br>`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.<br>• Not setting this key pair means that the trustpoint uses the default RSA key in the current configuration. |
| **Step 8** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Authenticating the CA

This task authenticates the CA to your router.

The router must authenticate the CA by obtaining the self-signed certificate of the CA, which contains the public key of the CA. Because the certificate of the CA is self-signed (the CA signs its own certificate), manually authenticate the public key of the CA by contacting the CA administrator to compare the fingerprint of the CA certificate.

## SUMMARY STEPS

1. **crypto ca authenticate ca-name**
2. show crypto ca certificates

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **crypto ca authenticate ca-name**<br>**Example:**<br>`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<br>**Example:**<br>`RP/0/RP0/CPU0:router# show crypto ca certificates` | (Optional) Displays information about the CA certificate. |

# Requesting Your Own Certificates

This task requests certificates from the CA.

You must obtain a signed certificate from the CA for each of your router's RSA key pairs. If you generated general-purpose RSA keys, your router has only one RSA key pair and needs only one certificate. If you previously generated special usage RSA keys, your router has two RSA key pairs and needs two certificates.

**SUMMARY STEPS**

1. **crypto ca enroll ca-name**
2. **show crypto ca certificates**

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **crypto ca enroll ca-name**<br>**Example:**<br>`RP/0/RP0/CPU0:router# crypto ca enroll myca` | Requests certificates for all of your RSA key pairs.<br>• 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.<br>• 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.<br>• 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**<br>**Example:** | (Optional) Displays information about the CA certificate. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router# show crypto ca certificates` | |

# Configuring Certificate Enrollment Using Cut-and-Paste

This task declares the trustpoint certification authority (CA) that your router should use and configures that trustpoint CA for manual enrollment by using cut-and-paste.

### SUMMARY STEPS

1. **configure**
2. **crypto ca trustpoint** *ca-name*
3. enrollment terminal
4. Use the **commit** or **end** command.
5. **crypto ca authenticate** *ca-name*
6. **crypto ca enroll** *ca-name*
7. **crypto ca import** ca- *name* **certificate**
8. show crypto ca certificates

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **crypto ca trustpoint** *ca-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# crypto ca trustpoint myca RP/0/RP0/CPU0:router(config-trustp)#` | Declares the CA that your router should use and enters trustpoint configuration mode.<br>• Use the *ca-name* argument to specify the name of the CA. |
| **Step 3** | enrollment terminal<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-trustp)# enrollment terminal` | Specifies manual cut-and-paste certificate enrollment. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 5** | **crypto ca authenticate** *ca-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router# crypto ca authenticate myca` | Authenticates the CA by obtaining the certificate of the CA.<br>• Use the *ca-name* argument to specify the name of the CA. Use the same name that you entered in Step 2, on page 83. |
| **Step 6** | **crypto ca enroll** *ca-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router# crypto ca enroll myca` | Obtains the certificates for your router from the CA.<br>• 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**<br>**Example:**<br>`RP/0/RP0/CPU0:router# crypto ca import myca certificate` | Imports a certificate manually at the terminal.<br>• Use the *ca-name* argument to specify the name of the CA. Use the same name that you entered in Step 2.<br>**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<br>**Example:**<br>`RP/0/RP0/CPU0:router# show crypto ca certificates` | Displays information about your certificate and the CA certificate. |

# Configuration Examples for Implementing Certification Authority Interoperability

This section provides the following configuration example:

## Configuring Certification Authority Interoperability: Example

The following example shows how to configure CA interoperability.

Comments are included within the configuration to explain various commands.

```
configure
hostname myrouter
domain name mydomain.com
end
```

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

crypto key generate rsa mykey

The name for the keys will be:mykey
Choose the size of the key modulus in the range of 360 to 2048 for your General Purpose
Keypair
Choosing a key modulus greater than 512 may take a few minutes.
How many bits in the modulus [1024]:
Generating RSA keys ...
Done w/ crypto generate keypair
[OK]

show crypto key mypubkey rsa

Key label:mykey
Type     :RSA General purpose
Size     :1024
Created  :17:33:23 UTC Thu Sep 18 2003
Data     :
 30819F30 0D06092A 864886F7 0D010101 05000381 8D003081 89028181 00CB8D86
 BF6707AA FD7E4F08 A1F70080 B9E6016B 8128004C B477817B BCF35106 BC60B06E
 07A417FD 7979D262 B35465A6 1D3B70D1 36ACAFBD 7F91D5A0 CFB0EE91 B9D52C69
 7CAF89ED F66A6A58 89EEF776 A03916CB 3663FB17 B7DBEBF8 1C54AF7F 293F3004
 C15B08A8 C6965F1E 289DD724 BD40AF59 E90E44D5 7D590000 5C4BEA9D B5020301
 0001

! The following commands declare a CA and configure a trusted point.

configure
crypto ca trustpoint myca
enrollment url http://xyz-ultra5
enrollment retry count 25
enrollment retry period 2
rsakeypair mykey
end

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

! The following command authenticates the CA to your router.

crypto ca authenticate myca

Serial Number  :01
Subject Name   :
cn=Root coax-u10 Certificate Manager,ou=HFR,o=Cisco Systems,l=San Jose,st=CA,c=US
Issued By      :
cn=Root coax-u10 Certificate Manager,ou=HFR,o=Cisco Systems,l=San Jose,st=CA,c=US
Validity Start :07:00:00 UTC Tue Aug 19 2003
Validity End   :07:00:00 UTC Wed Aug 19 2020
Fingerprint:58 71 FB 94 55 65 D4 64 38 91 2B 00 61 E9 F8 05
Do you accept this certificate?? [yes/no]:yes

! The following command requests certificates for all of your RSA key pairs.

crypto ca enroll myca

% Start certificate enrollment ...
% Create a challenge password. You will need to verbally provide this
  password to the CA Administrator in order to revoke your certificate.
% For security reasons your password will not be saved in the configuration.
% Please make a note of it.
```

```
Password:
Re-enter Password:
    Fingerprint: 17D8B38D ED2BDF2E DF8ADBF7 A7DBE35A

! The following command displays information about your certificate and the CA certificate.

show crypto ca certificates

Trustpoint       :myca
============================================================
CA certificate
  Serial Number  :01
  Subject Name   :
        cn=Root coax-u10 Certificate Manager,ou=HFR,o=Cisco Systems,l=San Jose,st=CA,c=US
  Issued By      :
        cn=Root coax-u10 Certificate Manager,ou=HFR,o=Cisco Systems,l=San Jose,st=CA,c=US
  Validity Start :07:00:00 UTC Tue Aug 19 2003
  Validity End   :07:00:00 UTC Wed Aug 19 2020
Router certificate
  Key usage      :General Purpose
  Status         :Available
  Serial Number  :6E
  Subject Name   :
        unstructuredName=myrouter.mydomain.com,o=Cisco Systems
  Issued By      :
        cn=Root coax-u10 Certificate Manager,ou=HFR,o=Cisco Systems,l=San Jose,st=CA,c=US
  Validity Start :21:43:14 UTC Mon Sep 22 2003
  Validity End   :21:43:14 UTC Mon Sep 29 2003
  CRL Distribution Point
        ldap://coax-u10.cisco.com/CN=Root coax-u10 Certificate Manager,O=Cisco Systems
```

# Where to Go Next

After you have finished configuring CA interoperability, you should configure IKE, IPSec, and SSL. IPSec in the *Implementing IPSec Network Security on* the Cisco IOS XR Softwaremodule, and SSL in the *Implementing Secure Socket Layer on* the Cisco IOS XR Softwaremodule. These modules are located in *System Security Configuration Guide for Cisco CRS Routers* (this publication).

# Additional References

The following sections provide references related to implementing certification authority interoperability.

### Related Documents

| Related Topic | Document Title |
|---|---|
| PKI commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *Public Key Infrastructure Commands on* the Cisco IOS XR Software module in *System Security Command Reference for Cisco CRS Routers*. |

### Standards

| Standards | Title |
|---|---|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

### MIBs

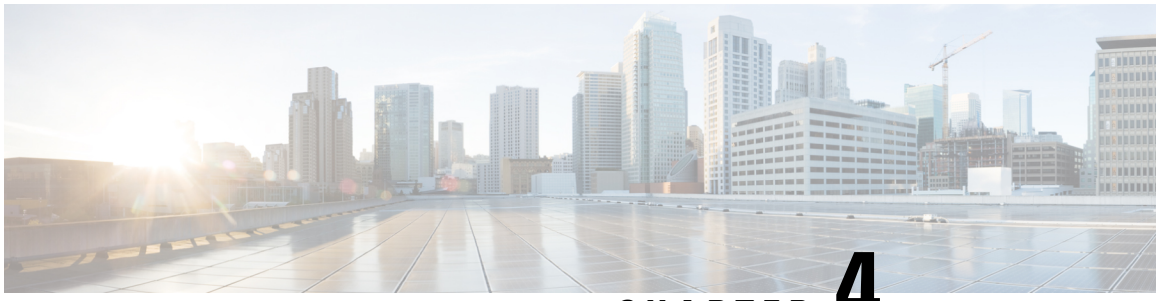
| MIBs | MIBs Link |
|---|---|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml |

### RFCs

| RFCs | Title |
|---|---|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 4

# Implementing Keychain Management

This module describes how to implement keychain management on. Keychain management is a common method of authentication to configure shared secrets on all entities that exchange secrets such as keys, before establishing trust with each other. Routing protocols and network management applications on Cisco IOS XR software often use authentication to enhance security while communicating with peers.

**Feature History for Implementing Keychain Management**

| Release | Modification |
|---|---|
| Release 3.3.0 | This feature was introduced. |
| Release 3.4.0 | • Support for the MAC authentication algorithm was added.<br>• Support for hitless key rollover and key acceptance tolerance were added. |
| Release 3.5.0 | Support for hitless key rollover for Open Shortest Path First (OSPF) and Intermediate System-to-Intermediate System (IS-IS) was added. |



# Prerequisites for Configuring Keychain Management

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

# Restrictions for Implementing Keychain Management

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

# Information About Implementing Keychain Management

The keychain by itself has no relevance; therefore, it must be used by an application that needs to communicate by using the keys (for authentication) with its peers. The keychain provides a secure mechanism to handle the keys and rollover based on the lifetime. Border Gateway Protocol (BGP), Open Shortest Path First (OSPF), and Intermediate System-to-Intermediate System (IS-IS) use the keychain to implement a hitless key rollover for authentication. BGP uses TCP authentication, which enables the authentication option and sends the Message Authentication Code (MAC) based on the cryptographic algorithm configured for the keychain. For information about BGP, OSPF, and IS-IS keychain configurations, see *Routing Configuration Guide for Cisco CRS Routers*

To implement keychain management, you must understand the concept of key lifetime, which is explained in the next section.

## Lifetime of Key

If you are using keys as the security method, you must specify the lifetime for the keys and change the keys on a regular basis when they expire. To maintain stability, each party must be able to store and use more than one key for an application at the same time. A keychain is a sequence of keys that are collectively managed for authenticating the same peer, peer group, or both.

Keychain management groups a sequence of keys together under a keychain and associates each key in the keychain with a lifetime.

**Note** Any key that is configured without a lifetime is considered invalid; therefore, the key is rejected during configuration.

The lifetime of a key is defined by the following options:

- Start-time—Specifies the absolute time.
- End-time—Specifies the absolute time that is relative to the start-time or infinite time.

Each key definition within the keychain must specify a time interval for which that key is activated; for example, lifetime. Then, during a given key's lifetime, routing update packets are sent with this activated key. Keys cannot be used during time periods for which they are not activated. Therefore, we recommend that for a given keychain, key activation times overlap to avoid any period of time for which no key is activated. If a time period occurs during which no key is activated, neighbor authentication cannot occur; therefore, routing updates can fail.

Multiple keychains can be specified.

# How to Implement Keychain Management

This section contains the following procedures:

# Configuring a Keychain

This task configures a name for the keychain.

You can create or modify the name of the keychain.

### SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. Use the **commit** or **end** command.
4. **show key chain** *key-chain-name*

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **key chain** *key-chain-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# key chain isis-keys`<br>`RP/0/RP0/CPU0:router(config-isis-keys)#` | Creates a name for the keychain.<br>**Note** 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 global configuration mode attributes or keychain-key configuration mode attributes (for example, lifetime or key string). |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 4** | **show key chain** *key-chain-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router# show key chain isis-keys` | (Optional) Displays the name of the keychain.<br>**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. |

### What to do next

After completing keychain configuration, see the Configuring a Tolerance Specification to Accept Keys, on page 92 section.

# Configuring a Tolerance Specification to Accept Keys

This task configures the tolerance specification to accept keys for a keychain to facilitate a hitless key rollover for applications, such as routing and management protocols.

### SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **accept-tolerance** *value* [**infinite**]
4. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **key chain** *key-chain-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# key chain isis-keys` | Creates a name for the keychain. |
| **Step 3** | **accept-tolerance** *value* [**infinite**]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-keys)# accept-tolerance infinite` | Configures a tolerance value to accept keys for the keychain.<br>• Use the *value* argument to set the tolerance range in seconds. The range is from 1 to 8640000.<br>• Use the **infinite** keyword to specify that the tolerance specification is infinite. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Configuring a Key Identifier for the Keychain

This task configures a key identifier for the keychain.

You can create or modify the key for the keychain.

## SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*
4. Use the **commit** or **end** command.

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **key chain** *key-chain-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# key chain isis-keys` | Creates a name for the keychain. |
| **Step 3** | **key** *key-id*<br>**Example:**<br>`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.<br>• Use the *key-id* argument as a 48-bit integer. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

### What to do next

After configuring a key identifier for the keychain, see the Configuring the Text for the Key String, on page 94 section.

# Configuring the Text for the Key String

This task configures the text for the key string.

### SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*
4. **key-string** [**clear** | **password**] *key-string-text*
5. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **key chain** *key-chain-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# key chain isis-keys` | Creates a name for the keychain. |
| **Step 3** | **key** *key-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-keys)# key 8`<br>`RP/0/RP0/CPU0:router(config-isis-keys-0x8)#` | Creates a key for the keychain. |
| **Step 4** | **key-string** [**clear** | **password**] *key-string-text*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-keys-0x8)#`<br>`key-string password 8` | Specifies the text string for the key.<br>• Use the **clear** keyword to specify the key string in clear text form; use the **password** keyword to specify the key in encrypted form.<br>• For a string to be a valid password, it must comply with the following rules:<br>• |
| **Step 5** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| | | • **Cancel** —Remains in the configuration session, without committing the configuration changes. |

### What to do next

After configuring the text for the key string, see the Configuring the Keys to Generate Authentication Digest for the Outbound Application Traffic, on page 96 section.

# Determining the Valid Keys

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

### SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*
4. **accept-lifetime** *start-time* [**duration** *duration-value* | **infinite** | *end-time*]
5. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **key chain** *key-chain-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# key chain isis-keys` | Creates a a name for the keychain. |
| **Step 3** | **key** *key-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-keys)# key 8`<br>`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*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-keys)# key 8`<br>`RP/0/RP0/CPU0:router(config-isis-keys-0x8)#`<br>`accept-lifetime 1:00:00 october 24 2005 infinite` | (Optional) Specifies the validity of the key lifetime in terms of clock time. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

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

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

### SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*
4. **send-lifetime** *start-time* [**duration** *duration-value* | **infinite** | *end-time*]
5. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **key chain** *key-chain-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# key chain isis-keys` | Creates a name for the keychain. |
| **Step 3** | **key** *key-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-keys)# key 8`<br>`RP/0/RP0/CPU0:router(config-isis-keys-0x8)#` | Creates a key for the keychain. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **send-lifetime** *start-time* [**duration** *duration-value* \| **infinite** \| *end-time*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-keys)# key 8`<br>`RP/0/RP0/CPU0:router(config-isis-keys-0x8)#`<br>`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.<br>In addition, you can specify a start-time value and one of the following values:<br>• **duration** keyword (seconds)<br>• **infinite** keyword<br>• *end-time* argument<br>If you intend to set lifetimes on keys, Network Time Protocol (NTP) or some other time synchronization method is recommended. |
| **Step 5** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Configuring the Cryptographic Algorithm

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

From Cisco IOS XR Software Release 6.7.2 and later, you must follow the below guidelines while configuring the key chain. These are applicable only for FIPS mode (that is, when **crypto fips-mode** is configured).

- You must configure the session with a FIPS-approved cryptographic algorithm. A session configured with non-approved cryptographic algorithm for FIPS (such as, **MD5** and **HMAC-MD5**) does not work. This is applicable for OSPF, BGP, RSVP, ISIS, or any application using key chain with non-approved cryptographic algorithm.
- If you are using any **HMAC-SHA** algorithm for a session, then you must ensure that the configured *key-string* has a minimum length of 14 characters. Otherwise, the session goes down.

## SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*

4. **cryptographic-algorithm** [**HMAC-MD5** | **HMAC-SHA1-12** | **HMAC-SHA1-20** | **MD5** | **SHA-1** | **AES-128-CMAC-96** | **HMAC-SHA-256** | **HMAC-SHA1-96**]
5. Use the **commit** or **end** command.

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **key chain** *key-chain-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# key chain isis-keys`<br>`RP/0/RP0/CPU0:router(config-isis-keys)#` | Creates a name for the keychain. |
| **Step 3** | **key** *key-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-keys)# key 8`<br>`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**]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-isis-keys-0x8)#`<br>`cryptographic-algorithm MD5` | Specifies the choice of the cryptographic algorithm. You can choose from the following list of algorithms:<br>• HMAC-MD5<br>• HMAC-SHA1-12<br>• HMAC-SHA1-20<br>• MD5<br>• SHA-1<br>• HMAC-SHA-256<br>• HMAC-SHA1-96<br>• AES-128-CMAC-96<br>The routing protocols each support a different set of cryptographic algorithms:<br>• Border Gateway Protocol (BGP) supports HMAC-MD5, HMAC-SHA1-12, HMAC-SHA1-96 and AES-128-CMAC-96.<br>• Intermediate System-to-Intermediate System (IS-IS) supports HMAC-MD5, SHA-1, MD5, AES-128-CMAC-96, HMAC-SHA-256, |

| | Command or Action | Purpose |
|---|---|---|
| | | HMAC-SHA1-12, HMAC-SHA1-20, and HMAC-SHA1-96.<br>• Open Shortest Path First (OSPF) supports MD5, HMAC-MD5, HMAC-SHA-256, HMAC-SHA1-12, HMAC-SHA1-20, and HMAC-SHA1-96. |
| **Step 5** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Configuration Examples for Implementing Keychain Management

This section provides the following configuration example:

## Configuring Keychain Management: Example

The following example shows how to configure keychain management:

```
configure
key chain isis-keys
accept-tolerance infinite
key 8
key-string mykey91abcd
cryptographic-algorithm MD5
send-lifetime 1:00:00 june 29 2006 infinite
accept-lifetime 1:00:00 june 29 2006 infinite
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]
```

# Additional References

The following sections provide references related to implementing keychain management.

### Related Documents

| Related Topic | Document Title |
|---|---|
| Keychain management commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *Keychain Management Commands in the System Security Command Reference for Cisco CRS Routers* |

### Standards

| Standards | Title |
|---|---|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

### MIBs

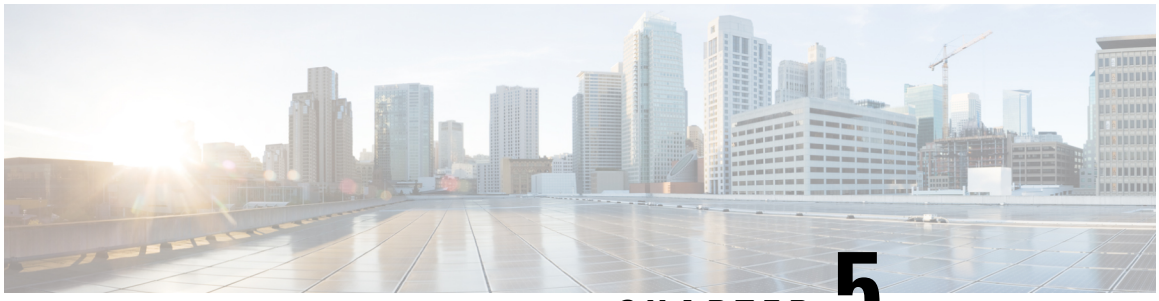
| MIBs | MIBs Link |
|---|---|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml |

### RFCs

| RFCs | Title |
|---|---|
| No new or modified RFCs are supported by this feature. | — |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 5

# Implementing Lawful Intercept

Lawful intercept is the process by which law enforcement agencies conduct electronic surveillance of circuit and packet-mode communications, authorized by judicial or administrative order. Service providers worldwide are legally required to assist law enforcement agencies in conducting electronic surveillance in both circuit-switched and packet-mode networks.

Only authorized service provider personnel are permitted to process and configure lawfully authorized intercept orders. Network administrators and technicians are prohibited from obtaining knowledge of lawfully authorized intercept orders, or intercepts in progress. Error messages or program messages for intercepts installed in the router are not displayed on the console.

Lawful Intercept is not a part of the Cisco IOS XR software by default. You have to install it separately by installing and activating **hfr-li.px.pie** .

### Feature History for Implementing Lawful Intercept

| Release | Modification |
|---|---|
| Release 3.8.0 | This feature was introduced. |
| Release 4.0.1 | Support for Lawful Intercept IPv6 on CRS-MSC-40G and CRS-FP-40 line cards was added. Information on intercepting IPv6 packets, lawful intercept filters, intercepting 6VPE and 6PE packets, IPv6 MD encapsulation, and per tap drop counter support was added. |
| Release 4.1.0 | Support for Lawful Intercept on CRS-MSC-140G and CRS-FP-140 line cards was added. |
| Release 4.2.0 | High Availability support for Lawful Intercept was added. |
| Release 4.3.2 | Lawful Intercept is available as a separate package. It is no longer a part of the Cisco IOS XR software. |

• Additional References, on page 121

# Prerequisites for Implementing Lawful Intercept

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Lawful intercept implementation also requires that these prerequisites are met:

- **Provisioned router—**The router must be already provisioned. For more information, see *Cisco IOS XR Getting Started Guide for the Cisco CRS Router*.

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

- **Understanding of SNMP Server commands in Cisco IOS XR software—**Simple Network Management Protocol, version 3 (SNMP v3), which is the basis for lawful intercept enablement, is configured using commands described in the module *SNMP Server Commands* in *System Management Command Reference for Cisco CRS Routers* . To implement lawful intercept, you must understand how the SNMP server functions. For this reason, carefully review the information described in the module *Implementing SNMP* in *System Management Configuration Guide for Cisco CRS Routers* .
- **Lawful intercept must be explicitly disabled—**It is automatically enabled on a provisioned router. However, you should not disable LI if there is an active tap in progress, because this deletes the tap.
- **Management plane configured to enable SNMPv3—**Allows the management plane to accept SNMP commands, so that the commands go to the interface (preferably, a loopback) on the router. This allows the mediation device (MD) to communicate with a physical interface.
- **VACM views enabled for SNMP server—**View-based access control model (VACM) views must be enabled on the router.
- **Provisioned MD—**For detailed information, see the vendor documentation associated with your MD. For a list of MD equipment suppliers preferred by Cisco, see http://www.cisco.com/en/US/tech/tk583/tk799/tsd_technology_support_protocol_home.html.
- **VoIP surveillance-specific requirements**
  - **Lawful-intercept-enabled call agent—**A lawful-intercept-enabled call agent must support interfaces for communications with the MD, for the target of interest to provide signaling information to the MD. The MD extracts source and destination IP addresses and Real-Time Protocol (RTP) port numbers from the Session Description Protocol (SDP) signaling information for the target of interest. It uses these to form an SNMPv3 SET, which is sent to the router acting as the content IAP to provision the intercept for the target of interest.

    The MD uses the CISCO-TAP2-MIB to set up communications between the router acting as the content IAP, and the MD.

    The MD uses the CISCO-IP-TAP-MIB to set up the filter for the IP addresses and port numbers to be intercepted and derived from the SDP.

- Routers to be used for calls by the target number must be provisioned for this purpose through the MD.
- The MD that has been provisioned with the target number to be intercepted.

- **Data session surveillance-specific requirements**
  - Routers to be used by the data target that have been provisioned for this purpose through the MD.
  - **The MD that has been provisioned with the user login ID, mac address of the user CPE device, or the DSLAM physical location ID—**The IP address is the binding that is most frequently used to identify the target in the network. However, alternative forms of information that uniquely identify the target in the network might be used in some network architectures. Such alternatives include the MAC address and the acct-session-id.

- The MD can be located anywhere in the network but must be reachable from the content IAP router, which is being used to intercept the target. MD should be reachable ONLY from global routing table and NOT from VRF routing table.

# Restrictions for Implementing Lawful Intercept

The following restrictions are applicable for Lawful Intercept:

- If lawful intercept is set up separately for two inter-communicating hosts with two different mediation devices, then by default, only the ingress traffic on the ASR 9000 router from one of the hosts is intercepted. You can configure the **overlap-tap enable** command to separately intercept the ASR 9000 ingress as well as egress traffic for both the mediation devices.
- Lawful intercept does not provide support for these features on Cisco CRS Router:
  - IPv6 multicast tapping
  - Per layer 3 interface tapping

**Note** Per layer 2 interface tapping is supported.

  - Replicating a single tap to multiple MDs
  - Tapping basic mpls packets
  - Tapping L2 flows
  - RTP encapsulation
  - GRE encapsulation
  - MPLS encapsulation

- Lawful intercept is applied only on ingress traffic.

  Traffic is intercepted, when it arrives as pure IP in the following scenarios:

  - For label imposition direction

- When it arrives from the core after PHP action.

Traffic is not intercepted in the following criteria:

- When it arrives from the core as MPLS encapsulated (with VPN label) for the label disposition direction.
- For GRE encapsulated packets.

# Information About Lawful Intercept Implementation

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

Lawful intercept offers these capabilities:

- Voice-over IP (VoIP) and data session intercept provisioning from the MD using SNMPv3
- Delivery of intercepted VoIP and data session data to the MD
- SNMPv3 lawful intercept provisioning interface
- Lawful intercept MIB: CISCO-TAP2-MIB, version 2
- CISCO-IP-TAP-MIB manages the Cisco intercept feature for IP and is used along with CISCO-TAP2-MIB to intercept IP traffic.
- User datagram protocol (UDP) encapsulation to the MD
- IPv6 MD encapsulation for modular services card and forwarding processor card (CRS-MSC-40G and CRS-FP-40)
- IPv6 MD encapsulation for modular services card and forwarding processor card (CRS-MSC-140G and CRS-FP-140)
- Replication and forwarding of intercepted packets to the MD
- Per tap drop counter support
- Voice-over IP (VoIP) call intercept, based on any rules configured for received packets.
- Voice-over IP (VoIP) intercept with LI-enabled call agent
- Data session call intercept based on IP address

## Interception Mode

The lawful intercept has two interception modes:

- **Global LI:** The taps are installed on all the line cards in the ingress direction. With the global tap, the traffic for the target can be intercepted regardless of ingress point. Only the tap that has wild cards in the interface field is supported.
- **Interface LI:** Taps each packet that is entering or leaving an interface without any additional filters.

# Overlapping Taps

Traffic interception can be configured for two inter-communicating intercepted hosts using overlapping taps.

For example, consider two taps, one configured for all traffic from source address A and another for all traffic going to destination address B. When a packet arrives with source address A and destination address B, the packet is tapped by TAP1 in ingress and TAP2 in egress, and copies will be generated and forwarded to both mediation devices. Overlapping taps can be enabled using **overlap-tap enable** command in Global configuration mode.

# Provisioning for VoIP Calls

Lawful Intercept provisioning for VoIP occurs in these ways:

- Security and authentication occurs because users define this through SNMPv3.
- The MD provisions lawful intercept information using SNMPv3.
- Network management occurs through standard MIBs.

# Call Interception

VoIP calls are intercepted in this manner:

- The MD uses configuration commands to configure the intercept on the call control entity.
- The call control entity sends intercept-related information about the target to the MD.
- The MD initiates call content intercept requests to the content IAP router or trunk gateway through SNMPv3.
- The content IAP router or trunk gateway intercepts the call content, replicates it, and sends it to the MD in Packet Cable Electronic Surveillance UDP format. Specifically, the original packet starting at the first byte of the IP header is prefixed with a four-byte CCCID supplied by the MD in TAP2-MIB. It is then put into a UDP frame with the destination address and port of the MD.
- After replicated VoIP packets are sent to the MD, the MD then forwards a copy to a law-enforcement-agency-owned collection function, using a recognized standard.

# Provisioning for Data Sessions

Provisioning for data sessions occurs in a similar way to the way it does for lawful intercept for VoIP calls. (See Provisioning for VoIP Calls, on page 105.)

# Data Interception

Data are intercepted in this manner:

- If a lawful intercept-enabled authentication or accounting server is not available, a sniffer device can be used to detect the presence of the target in the network.
    - The MD uses configuration commands to configure the intercept on the sniffer.

- The sniffer device sends intercept-related information about the target to the MD.

- The MD initiates communication content intercept requests to the content IAP router using SNMPv3.
- The content IAP router intercepts the communication content, replicates it, and sends it to the MD in UDP format.
- Intercepted data sessions are sent from the MD to the collection function of the law enforcement agency, using a supported delivery standard for lawful intercept.

### Information About the MD

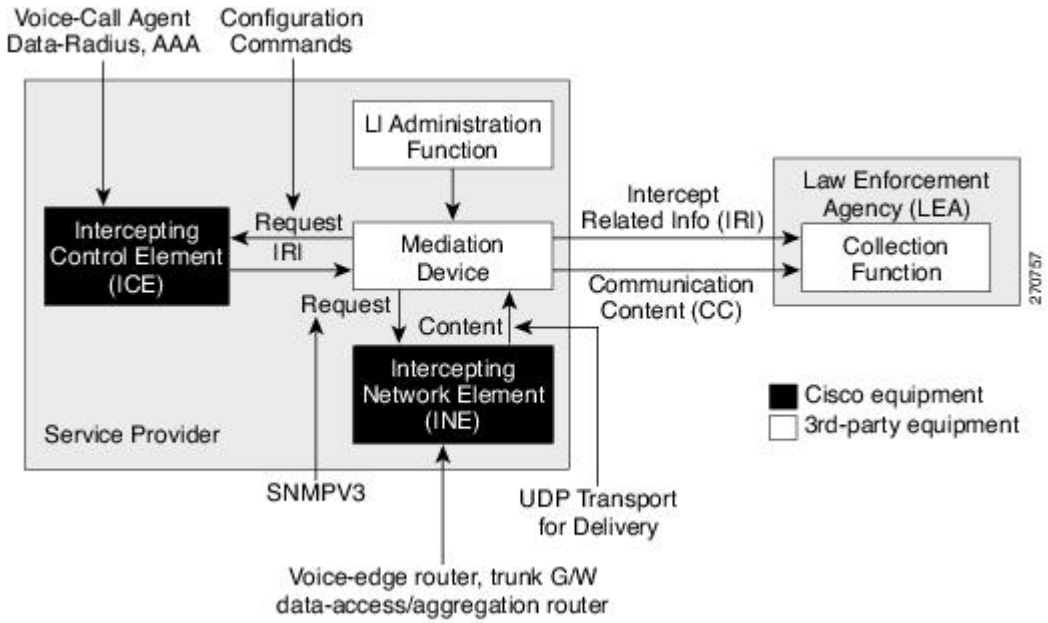
The MD performs these tasks:

- Activates the intercept at the authorized time and removes it when the authorized time period elapses.
- Periodically audits the elements in the network to ensure that:
  - *only* authorized intercepts are in place.
  - *all* authorized intercepts are in place.

# Lawful Intercept Topology

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

***Figure 1: Lawful Intercept Topology for Both Voice and Data Interception***



## Layer 2 Lawful Intercept

You can configure SNMP-based lawful intercept on a layer 2 interface. This intercepts all traffic passing through the particular interface.

## Scale or Performance Improvement

The new enhancement in terms of scalability and performance for lawful intercept support on Cisco CRS Router modular services cards and forwarding processor cards (CRS-MSC-40G, CRS-FP-40, CRS-MSC-140G, CRS-FP-140) are:

- Increase in IPv4 lawful intercept tap limit from 250 taps to 1000 taps.
- IPv4 and IPv6 combined tap limit is 2000 taps, 1000 taps each for IPv4 and IPv6.
- Enhancement of interception rate to 100 Mbps per slot.
- Enhancement of interception rate from 100 Mbps to 1 Gbps per slot.

# Intercepting IPv4 and IPv6 Packets

This section provides details for intercepting IPv4 and IPv6 packets supported on the Cisco CRS Router.

## Lawful Intercept Filters

The filters used for classifying a tap are:

- IP address type
- Destination address
- Destination mask
- Source address
- Source mask
- ToS (Type of Service) and ToS mask
- Protocol
- Destination port with range
- Source port with range
- VRF (VPN Routing and Forwarding)
- Flow ID

# Intercepting Packets Based on Flow ID (Applies to IPv6 only)

To further extend filteration criteria for IPv6 packets, an additional support to intercept IPv6 packets based on flow ID has been introduced on the Cisco CRS Router. All IPv6 packets are intercepted based on the fields in the IPv6 header which comprises numerous fields defined in IPv6 Header Field Details table:

**Note** The field length or payload length is not used for intercepting packets.

***Table 4: IPv6 Header Field Details***

| IPv6 Field Name | Field Description | Field Length |
|---|---|---|
| Version | IPv6 version number. | 4 bits |
| Traffic Class | Internet traffic priority delivery value. | 8 bits |
| Flow ID (Flow Label) | Used for specifying special router handling from source to destination(s) for a sequence of packets. | 20 bits |
| Payload Length | Specifies the length of the data in the packet. When cleared to zero, the option is a hop-by-hop Jumbo payload. | 16 bits unassigned |
| Next Header | Specifies the next encapsulated protocol. The values are compatible with those specified for the IPv4 protocol field. | 8 bits |
| Hop Limit | For each router that forwards the packet, the hop limit is decremented by 1. When the hop limit field reaches zero, the packet is discarded. This replaces the TTL field in the IPv4 header that was originally intended to be used as a time based hop limit. | 8 bits unsigned |
| Source Address | The IPv6 address of the sending node. | 16 bytes |
| Destination Address | The IPv6 address of the destination node. | 16 bytes |

The flow ID or flow label is a 20 bit field in the IPv6 packet header that is used to discriminate traffic flows. Each flow has a unique flow ID. The filteration criteria to intercept packets matching a particular flow ID is defined in the tap configuration file. From the line card, the intercepted mapped flow IDs are sent to the next hop, specified in the MD configuration file. The intercepted packets are replicated and sent to the MD fom the line card. The ideal replication rate on MSC-40 is 100 Mbps depending on packet size and features configured on the router.

# Intercepting VRF (6VPE) and 6PE Packets

This section provides information about intercepting VRF aware packets and 6PE packets. Before describing how it works, a basic understanding of 6VPE networks is discussed.

The MPLS VPN model is a true peer VPN model. It enforces traffic separations by assigning unique VPN route forwarding (VRF) tables to each customer's VPN at the provider content IAP router. Thus, users in a specific VPN cannot view traffic outside their VPN.

Cisco CRS Router supports intercepting IPv6 packets of the specified VRF ID for 6VPE. To distiguish traffic on VPN, VRFs are defined containing a specific VRF ID. The filter criteria to tap a particular VRF ID is specified in the tap. IPv6 packets are intercepted with the VRF context on both scenarios: imposition (ip2mpls) and disposition (mpls2ip).

The 6PE packets carry IPv6 packets over VPN. The packets do not have a VRF ID. Only IP traffic is intercepted; no MPLS based intercepts are supported. The IPv6 traffic is intercepted at the content IAP of the MPLS cloud at imposition (ip2mpls) and at disposition (mpls2ip).

Intercepting IPv6 packets is also performed for ip2tag and tag2ip packets. Ip2tag packets are those which are converted from IPv6 to Tagging (IPv6 to MPLS), and tag2ip packects are those which are converted from Tagging to IPv6 (MPLS to IPv6) at the provider content IAP router.

## Encapsulation Type Supported for Intercepted Packets

Intercepted packets mapping the tap are replicated, encapsulated, and then sent to the MD. IPv4 and IPv6 packets are encapsulated using UDP (User Datagram Protocol) encapsulation. The replicated packets are forwarded to MD using UDP as the content delivery protocol. Both IPv4 and IPv6 MD encapsulations are supported. The encapsulation type (IPv4 or IPv6) depends on the address of MD.

The intercepted packet gets a new UDP header and IPv4 header. Information for IPv4 header is derived from MD configuration. Apart from the IP and UDP headers, a 4 byte channel identifier (CCCID) is also inserted after the UDP header in the packet. After adding the MD encapsulation, if the packet size is above the MTU, the egress LC CPU fragments the packet. Moreover, there is a possibility that the packet tapped is already a fragment. Each tap is associated with only one MD. Cisco CRS Router does not support forwarding replicated packets to multiple MDs.

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

## Per Tap Drop Counter Support

Cisco CRS Router line cards provide SNMP server as an interface to export each tap forwarded to MD packet and drop counts. Any intercepted packets that are dropped prior to getting forwarded to the MD due to policer action are counted and reported. The drops due to policer action are the only drops that are counted under per tap drop counters. If a lawful intercept filter is modified, the packet counts are reset to 0.

# High Availability for Lawful Intercept

High availability for lawful intercept provides operational continuity of the TAP flows and provisioned MD tables to reduce loss of information due to route processor fail over (RPFO).

To achieve continuous interception of a stream, when RP fail over is detected; MDs are required to re-provision all the rows relating to CISCO-TAP2-MIB, CISCO-IP-TAP-MIB, and CISCO-USER-CONNECTION-TAP-MIB to synchronize database view across RP and MD.

**Note** The high availability for lawful intercept is enabled by default from Release 4.2.0 onwards.

## Preserving TAP and MD Tables during RP Fail Over

At any point in time, MD has the responsibility to detect the loss of the taps via SNMP configuration process.

After RPFO is completed, MD should re-provision all the entries in the stream tables, MD tables, and IP taps with the same values they had before fail over. As long as an entry is re-provisioned in time, existing taps will continue to flow without any loss.

The following restrictions are listed for re-provisioning MD and tap tables with respect to behavior of SNMP operation on citapStreamEntry, cTap2StreamEntry, cTap2MediationEntry MIB objects:

- After RPFO, table rows that are not re-provisioned, shall return NO_SUCH_INSTANCE value as result of SNMP Get operation.
- Entire row in the table must be created in a single configuration step, with exactly same values as before RPFO, and with the rowStatus as CreateAndGo. Only exception is the cTap2MediationTimeout object, that should reflect valid future time.

## Replay Timer

The replay timer is an internal timeout that provides enough time for MD to re-provision tap entries while maintaining existing tap flows. It resets and starts on the active RP when RPFO takes place. The replay timer is a factor of number of LI entries in router with a minimum value of 10 minutes.

After replay timeout, interception stops on taps that are not re-provisioned.

**Note** In case high availability is not required, MD waits for entries to age out after fail over. MD cannot change an entry before replay timer expiry. It can either reinstall taps as is, and then modify; or wait for it to age out.

# Installing Lawful Intercept (LI) Package

As LI is not a part of the Cisco IOS XR image by default, you need to install it separately.

## Installing and Activating the LI Package

The Package Installation Envelope (PIE) files, are installable software files with the .pie extension. PIE files are used to copy one or more software components onto the router. A PIE may contain a single component, a group of components (called a package), or a set of packages (called a composite package).

Use the **show install committed** command in EXEC mode to verify the committed software packages.

To install the Lawful Intercept (LI) package, you must install and activate the **hfr-li-px.pie**

For more information about installing PIEs, refer to *Upgrading and Managing Cisco IOS XR Software section* of the *System Management Configuration Guide for Cisco CRS Routers* .

**SUMMARY STEPS**

1. **admin**

2. **install add** *tftp://<IP address of tftp server>/<location of pie on server>*
3. **install activate** *device:package*
4. **install commit**
5. **exit**
6. **show install committed**

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **admin**<br>**Example:**<br>`RP/0/RP0/CPU0:router# admin` | Enters administration EXEC mode. |
| **Step 2** | **install add** *tftp://<IP address of tftp server>/<location of pie on server>*<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install add tftp://172.201.11.140/auto/tftp-users1/`**`hfr-li-px.pie`** | Copies the contents of a package installation envelope (PIE) file to a storage device. |
| **Step 3** | **install activate** *device:package*<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install activate disk0:hfr-li-px.pie` | Activates the respective package and adds more functionality to the existing software. |
| **Step 4** | **install commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install commit` | Saves the active software set to be persistent across designated system controller (DSC) reloads. |
| **Step 5** | **exit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# exit` | Exits from the admin mode. |
| **Step 6** | **show install committed**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show install committed` | Shows the list of the committed software packages. |

# Deactivating the LI PIE

To uninstall the Lawful Intercept package, deactivate hfr-li-px.pie as shown in the following steps:

**Note** You might experience interface or protocol flaps while uninstalling or deactivating the LI PIE. Hence, we recommend you to perform this activity during a maintenance window.

**SUMMARY STEPS**

1. **admin**
2. **install deactivate** *device:package*
3. **install commit**
4. **install remove** *device:package*
5. **exit**
6. **show install committed**

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **admin**<br>**Example:**<br>`RP/0/RP0/CPU0:router# admin` | Enters administration EXEC mode. |
| **Step 2** | **install deactivate** *device:package*<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install deactivate disk0:hfr-li-px.pie` | Activates the respective package and adds more functionality to the existing software. |
| **Step 3** | **install commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install commit` | Saves the active software set to be persistent across designated system controller (DSC) reloads. |
| **Step 4** | **install remove** *device:package*<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install remove disk0:hfr-li-px.pie` | Saves the active software set to be persistent across designated system controller (DSC) reloads. |
| **Step 5** | **exit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# exit` | Exits from the admin mode. |
| **Step 6** | **show install committed**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show install committed` | Shows the list of the committed software packages. |

# Upgrade and Downgrade Scenarios for the Lawful Intercept package

This section describes the possible upgrade and downgrade scenarios with respect to the Lawful Intercept (LI) package.

This example configuration demonstrates how to upgrade or downgrade the Cisco IOS XR software with or without the LI package. Suppose you have two versions of software images, V1 and V2. If you want to upgrade or downgrade from V1 to V2 without the LI package, you need to perform the following steps for the upgrade or the downgrade procedure:

**Note** Ensure that you use Turbo Boot to load the image for the downgrade process.

1. Ensure that the device has booted with the V1 image. Check the Package Installation Envelope (PIE) files that have been installed in V1.
2. Save all the PIE files that exist in V2 in the Trivial File Transfer Protocol (TFTP) server. Copy the contents of the PIE files from the TFTP server by using the **install add** command in the admin mode.

   ```
   RP/0/RSP0/CPU0:router(admin)# install add tar
   tftp://223.255.254.254/install/files/pies.tar
   ```

3. To activate all the PIE files in V2 at once, run the following commands based on the type of upgrade:
   - Normal Upgrade

     ```
     RP/0/RSP0/CPU0:router(admin)# install activate disk0:hfr-mini-px-4.3.99
     disk0:hfr-mpls-px-4.3.99 disk0:hfr-mgbl-px-4.3.99 disk0:hfr-mcast-px-4.3.99
     disk0:hfr-li-px-4.3.99 async
     ```

   - Cisco In-Service Software Upgrade (ISSU)

     ```
     RP/0/RSP0/CPU0:router(admin)# install activate disk0:hfr-mini-px-4.3.99
     disk0:hfr-mpls-px-4.3.99 disk0:hfr-mgbl-px-4.3.99 disk0:hfr-mcast-px-4.3.99
     disk0:hfr-li-px-4.3.99 async
     ```

At any point during the upgrade or the downgrade process, you can check the progress by using the **show install request** or the **show issu** command.

Some of the conventions that are followed in describing these scenarios are:

- Release 4.3.1 base image: It is the Cisco IOS XR software for Release 4.3.1 that contains Cisco LI by default.
- Release 4.3.2 base image: It is the Cisco IOS XR software for Release 4.3.2 that does not contain Cisco LI by default.
- Separate LI package: It is the LI package that needs to be installed separately for Release 4.3.2 and higher versions.

***Table 5: Upgrade Scenarios***

| Upgrade From | Upgrade To | Result | Supported |
|---|---|---|---|
| Release 4.3.1 base image | Release 4.3.2 base image | Before the upgrade, the LI has to be configured and provisioned completely. After the upgrade to Release 4.3.2 version without the LI package, you cannot configure or provision LI. | Yes |

| Upgrade From | Upgrade To | Result | Supported |
|---|---|---|---|
| Release 4.3.1 base image | Release 4.3.2 base image with the separate LI package | The Upgrade will reload the router. After the upgrade process completes, you need to reconfigure LI MDs/TAPs from the SNMP server. Also, all the LI configurations made in the earlier version is accepted. | Yes |
| Release 4.3.2 base image with the separate LI package | Release 4.3.3 base image with the separate LI package | After the upgrade, the LI configuration is not retained. | Yes |
| Release 4.3.2 base image with the separate LI package | Release 4.3.3 base image without the separate LI package | This upgrade is not possible as the installation process will not proceed without the LI PIE. | No |
| Release 4.3.2 base image without the separate LI package | Release 4.3.3 base image with the separate LI package | This upgrade is possible. | Yes |
| ISSU for Release 4.3.1 base image | Release 4.3.2 with the separate LI package | After this upgrade, to retain the LI configuration, you have to replay the configuration before the replay timeout occurs. | Yes |
| ISSU for Release 4.3.2 base image with the separate LI package | Release 4.3.3 with the separate LI package | After this upgrade, to retain the LI configuration, you have to replay the configuration before the replay timeout occurs. | Yes |

***Table 6: Downgrade Scenarios***

| Downgrade From | Downgrade To | Result | Supported |
|---|---|---|---|
| Release 4.3.2 base image without the separate LI package | Release 4.3.1 base image | After the downgrade, begin the provisioning process of LI. | Yes |
| Release 4.3.2 base image with the separate LI package | Release 4.3.1 base image | This scenario is not supported. | No |

| Downgrade From | Downgrade To | Result | Supported |
|---|---|---|---|
| Release 4.3.3 base image with the separate LI package | Release 4.3.2 base image with the separate LI package | After the downgrade, the LI configuration is not retained. You have to provision the LI once again. | Yes |
| Release 4.3.3 base image with the separate LI package | Release 4.3.2 base image without the LI package | After the downgrade, the LI configuration is lost. You will not be able to provision it after downgrade. | Yes |
| Release 4.3.3 base image | Release 4.3.2 base image with the separate LI package | The LI configuration is accepted and can be provisioned only after the downgrade. | Yes |
| ISSU | | | No |

# How to Configure SNMPv3 Access for Lawful Intercept

Perform these procedures in the order presented to configure SNMPv3 for the purpose of Lawful Intercept enablement:

## Disabling SNMP-based Lawful Intercept

Lawful Intercept is enabled by default on the  after installing and activating the .

- To disable Lawful Intercept, enter the **lawful-intercept disable** command in global configuration mode.
- To re-enable it, use the **no** form of this command.

### Disabling SNMP-based Lawful Intercept: Example

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# lawful-intercept disable
```

**Note** The **lawful-intercept disable** command is available on the router, only after installing and activating the .

All SNMP-based taps are dropped when lawful intercept is disabled.

# Configuring the Inband Management Plane Protection Feature

If MPP was not earlier configured to work with another protocol, then ensure that the MPP feature is also not configured to enable the SNMP server to communicate with the mediation device for lawful interception. In such cases, MPP must be configured specifically as an inband interface to allow SNMP commands to be accepted by the router, using a specified interface or all interfaces.

**Note** Ensure this task is performed, even if you have recently migrated to Cisco IOS XR Software from Cisco IOS, and you had MPP configured for a given protocol.

For lawful intercept, a loopback interface is often the choice for SNMP messages. If you choose this interface type, you must include it in your inband management configuration.

For a more detailed discussion of the inband management interface, see the Inband Management Interface, on page 124.

### Related Tasks

- Configuring a Device for Management Plane Protection for an Inband Interface, on page 126

### Related Examples

- Configuring the Inband Management Plane Protection Feature: Example, on page 120

# Enabling the Mediation Device to Intercept VoIP and Data Sessions

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

### SUMMARY STEPS

1. **configure**
2. **snmp-server view** *view-name* **ciscoTap2MIB included**
3. **snmp-server view** *view-name* **ciscoUserConnectionTapMIB included**
4. **snmp-server group** *group-name* **v3auth read** *view-name* **write** *view-name* **notify** *view-name*
5. **snmp-server host** *ip-address* **traps version 3 auth** *username* **udp-port** *port-number*
6. **snmp-server user** *mduser-id groupname* **v3 auth md5** md-*password*
7. Use the **commit** or **end** command.
8. show snmp users
9. show snmp group
10. show snmp view

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **snmp-server view** *view-name* **ciscoTap2MIB included**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server view TapName ciscoTap2MIB included` | Creates or modifies a view record and includes the CISCO-TAP2-MIB family in the view. The SNMP management objects in the CISCO-TAP2-MIB that controls lawful intercepts are included. This MIB is used by the mediation device to configure and run lawful intercepts on targets sending traffic through the router. |
| **Step 3** | **snmp-server view** *view-name* **ciscoUserConnectionTapMIB included**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server view TapName ciscoUserConnectionTapMIB included` | Creates or modifies a view record and includes the CISCO-USER-CONNECTION-TAP-MIB family, to manage the Cisco intercept feature for user connections. This MIB is used along with the CISCO-TAP2-MIB to intercept and filter user traffic. |
| **Step 4** | **snmp-server group** *group-name* **v3auth read** *view-name* **write** *view-name* **notify** *view-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server group TapGroup v3 auth read TapView write TapView notify TapView` | Configures a new SNMP group that maps SNMP users to SNMP views. This group must have read, write, and notify privileges for the SNMP view. |
| **Step 5** | **snmp-server host** *ip-address* **traps version 3 auth** *username* **udp-port** *port-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server host 223.255.254.224 traps version 3 auth bgreen udp-port 2555` | Specifies SNMP trap notifications, the version of SNMP to use, the security level of the notifications, and the recipient (host) of the notifications. |
| **Step 6** | **snmp-server user** *mduser-id groupname* **v3 auth md5** md-*password*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server mduser-id TapGroup v3 auth md5 mdpassword` | Configures the MD user as part of an SNMP group, using the v3 security model and the HMAC MD5 algorithm, which you associate with the MD password.<br>• The *mduser-id* and *mdpassword* must match that configured on MD. Alternatively, these values must match those in use on the router.<br>• Passwords must be eight characters or longer to comply with SNMPv3 security minimums.<br>• Minimum Lawful Intercept security level is auth; The noauth option will not work, as it indicates |

| | Command or Action | Purpose |
|---|---|---|
| | | noAuthnoPriv security level. The Lawful Intercept security level must also match that of the MD.<br>• Choices other than MD5 are available on the router, but the MD values must match.<br>Most MDs default to or support only MD5. |
| **Step 7** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 8** | show snmp users<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp users` | Displays information about each SNMP username in the SNMP user table. |
| **Step 9** | show snmp group<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp group` | Displays information about each SNMP group on the network. |
| **Step 10** | show snmp view<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp view` | Displays information about the configured views, including the associated MIB view family name, storage type, and status. |

# Adding MD and TAP Objects

To keep the MD row in active state, the following objects are mandatory:

- cTap2MediationDestAddressType
- cTap2MediationDestAddress
- cTap2MediationDestPort
- cTap2MediationSrcInteraface
- cTap2MediationTimeout
- cTap2MediationTransport

- cTap2MediationStatus

### SUMMARY STEPS

1. Add MD.
2. Add TAP.
3. Activate TAP.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | Add MD.<br>**Example:**<br>`setany -v3 <ip-address> <user>`<br>`cTap2MediationDestAddressType.1 <ipv4/ipv6>`<br>`cTap2MediationDestAddress.1 <"ip">`<br>`cTap2MediationDestPort.1 "1234"`<br>`cTap2MediationSrcInterface.1 0`<br>`cTap2MediationTransport.1 udp`<br>`cTap2MediationNotificationEnable.1 true`<br>`cTap2MediationTimeout.1 '7 de 6 14 3 4 5 6 2d 1 2'`<br>`cTap2MediationStatus.1 createAndGo`<br>`cTap2MediationDestAddressType.1 = ipv4(1)`<br>`cTap2MediationDestAddress.1 = 46 01  01 02`<br>`cTap2MediationDestPort.1 = 1234`<br>`cTap2MediationSrcInterface.1 = 0`<br>`cTap2MediationTransport.1 = udp(1)`<br>`cTap2MediationNotificationEnable.1 = true(1)`<br>`cTap2MediationTimeout.1 =`<br>`2014-Jun-20,03:04:05.6,-1:2`<br>`cTap2MediationStatus.1 = createAndGo(4)` | Creates an MD for mediation services.<br>To delete a MD, run:<br>`setany -v3 <ip-address> <user>`<br>`cTap2MediationStatus.1 6`<br>`cTap2MediationStatus.1 = destroy(6)` |
| **Step 2** | Add TAP.<br>**Example:**<br>`setany -v3 <ip-address> <user>`<br>`citapStreamInterface.1.1 0 citapStreamAddrType.1.1`<br>`<ipv4/ipv6>`<br>`citapStreamSourceAddress.1.1 "5a 1 1 2"`<br>`citapStreamSourceLength.1.1 32`<br>`citapStreamStatus.1.1 citapStreamInterface.1.1 =`<br>`0`<br>`citapStreamAddrType.1.1 = ipv4(1)`<br>`citapStreamSourceAddress.1.1 = 5a 01  01 02`<br>`citapStreamSourceLength.1.1 = 32`<br>`citapStreamStatus.1.1 = createAndGo(4)` | Creates a TAP for stream operation.<br>To delete a TAP, run:<br>`setany -v3 <ip-address> <user>`<br>`citapStreamStatus.1.1 6 cTap2StreamStatus.1.1 6`<br>`citapStreamStatus.1.1 = destroy(6)`<br>`cTap2StreamStatus.1.1 = destroy(6)` |
| **Step 3** | Activate TAP.<br>**Example:**<br>`setany -v3 <ip-address> <user> cTap2StreamType.1.1`<br>`ip cTap2StreamInterceptEnable.1.1 true`<br>`cTap2StreamStatus.1.1 createAndGo` | Activates the TAP for stream operation. |

| | Command or Action | Purpose |
|---|---|---|
| | cTap2StreamType.1.1 = ip(1)<br>cTap2StreamInterceptEnable.1.1 = true(1)<br>cTap2StreamStatus.1.1 = createAndGo(4)<br>**Example:**<br>To add TAP for L2VPN networks<br>`setany -v3 <ip-address> <user>`<br>`citapStreamInterface.4.1200 1125`<br>`citapStreamStatus.4.1200 createAndGo` | |

# Configuration Example for Inband Management Plane Feature Enablement

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

## Configuring the Inband Management Plane Protection Feature: Example

You must specifically enable management activities, either globally or on a per-inband-port basis, using this procedure. To globally enable inbound MPP, use the keyword **all** with the **interface** command, rather than use a particular interface type and instance ID with it.

```
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# control-plane
RP/0/RP0/CPU0:router(config-ctrl)# management-plane
RP/0/RP0/CPU0:router(config-mpp)# inband
RP/0/RP0/CPU0:router(config-mpp-inband)# interface loopback0
RP/0/RP0/CPU0:router(config-mpp-inband-Loopback0)# allow snmp
RP/0/RP0/CPU0:router(config-mpp-inband-Loopback0)# commit
RP/0/RP0/CPU0:router(config-mpp-inband-Loopback0)# exit
RP/0/RP0/CPU0:router(config-mpp-inband)# exit
RP/0/RP0/CPU0:router(config-mpp)# exit
RP/0/RP0/CPU0:router(config-ctr)# exit
RP/0/RP0/CPU0:router(config)# exit
RP/0/RP0/CPU0:router# show mgmt-plane inband interface loopback0

Management Plane Protection - inband interface


interface - Loopback0
        snmp configured -
                All peers allowed
RP/0/RP0/CPU0:router(config)# commit
```

# Additional References

These sections provide references related to implementing lawful intercept.

### Related Documents

| Related Topic | Document Title |
|---|---|
| Lawful Intercept commands | *System Security Command Reference for Cisco CRS Routers* |
| Implementing SNMP | *System Management Configuration Guide for Cisco CRS Routers* |
| SNMP Server commands | *System Management Command Reference for Cisco CRS Routers* |

### Standards

| Standards | Title |
|---|---|
| A modular, open architecture designed for simple implementation that easily interacts with third-party equipment to meet service provider lawful intercept requirements. | See RFC-3924 under RFCs, on page 121. |
| An application layer protocol that facilitates the exchange of management information between network devices. Part of the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite. | Simple Network Management Protocol Version 3 (SNMPv3) |

### MIBs

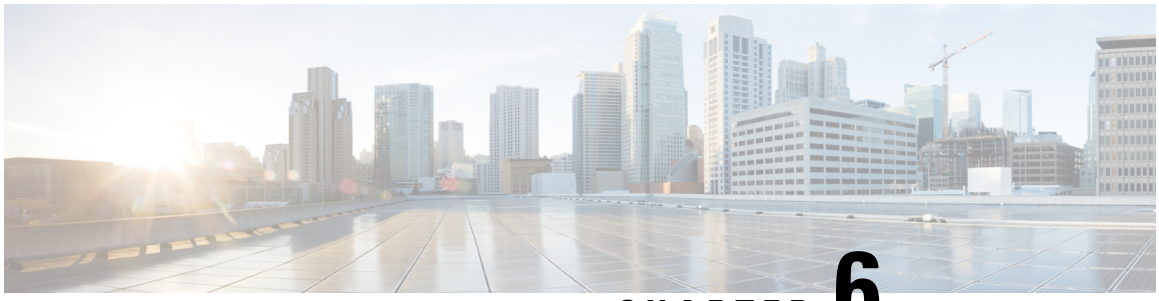
| MIBs | MIBs Link |
|---|---|
| • CISCO-TAP2-MIB, version 2<br>• CISCO-IP-TAP-MIB | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml |

### RFCs

| RFCs | Title |
|---|---|
| RFC-3924 | Cisco Architecture for Lawful Intercept in IP Networks |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access more content. | http://www.cisco.com/techsupport |



CHAPTER 6

# Implementing Management Plane Protection

The Management Plane Protection (MPP) feature in Cisco IOS XR software provides the capability to restrict the interfaces on which network management packets are allowed to enter a device. The MPP feature allows a network operator to designate one or more router interfaces as management interfaces.

Device management traffic may enter a device only through these management interfaces. After MPP is enabled, no interfaces except designated management interfaces accept network management traffic destined to the device. Restricting management packets to designated interfaces provides greater control over management of a device, providing more security for that device.

For information on MPP commands, see the *Management Plane Protection Commands* module in *System Security Command Reference for Cisco CRS Routers* .

### Feature History for Implementing Management Plane Protection

| Release | Modification |
|---|---|
| Release 3.5.0 | This feature was introduced. |
| Release 3.6.0 | The following enhancements were added:<br>• Out-of-band management interface support for applications.<br>• Peer-filtering for specific peers or a range of peers for the specified application. |
| Release 3.7.0 | The following enhancements were added:<br>• The information describing the MPP feature was expanded.<br>• New information was added about how logical and management plane interfaces filter packets based on the ingress physical interface. |

# Prerequisites for Implementing Management Plane Protection

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

# Restrictions for Implementing Management Plane Protection

The following restrictions are listed for implementing Management Plane Protection (MPP):

- Currently, MPP does not keep track of the denied or dropped protocol requests.
- MPP configuration does not enable the protocol services. MPP is responsible only for making the services available on different interfaces. The protocols are enabled explicitly.
- Management requests that are received on inband interfaces are not necessarily acknowledged there.
- Both Route Processor (RP) and distributed route processor (DRP) Ethernet interfaces are by default out-of-band interfaces and can be configured under MPP.
- The changes made for the MPP configuration do not affect the active sessions that are established before the changes.
- Currently, MPP controls only the incoming management requests for protocols, such as TFTP, Telnet, Simple Network Management Protocol (SNMP), Secure Shell (SSH), and HTTP.
- MPP does not support MIB.
- In a MPLS L3VPN, when MPP has VRF interface attached, it applies the VRF filter on an incoming interface through LPTS. When an incoming packet from the core interface has a different VRF, then MPP does not allow it.

**Note** When configuring a device for MPP for an inband interface the **Interface all** configuration does not apply specific VRF filter and allows traffic for all source and destination interfaces.

# Information About Implementing Management Plane Protection

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

## Inband Management Interface

An *inband management interface* is a Cisco IOS XR software physical or logical interface that processes management packets, as well as data-forwarding packets. An inband management interface is also called a *shared management interface*.

# Out-of-Band Management Interface

*Out-of-band* refers to an interface that allows only management protocol traffic to be forwarded or processed. An *out-of-band management interface* is defined by the network operator to specifically receive network management traffic. The advantage is that forwarding (or customer) traffic cannot interfere with the management of the router, which significantly reduces the possibility of denial-of-service attacks.

Out-of-band interfaces forward traffic only between out-of-band interfaces or terminate management packets that are destined to the router. In addition, the out-of-band interfaces can participate in dynamic routing protocols. The service provider connects to the router's out-of-band interfaces and builds an independent overlay management network, with all the routing and policy tools that the router can provide.

# Peer-Filtering on Interfaces

The peer-filtering option allows management traffic from specific peers, or a range of peers, to be configured.

# Control Plane Protection Overview

A *control plane* is a collection of processes that run at the process level on a route processor and collectively provide high-level control for most Cisco IOS XR software functions. All traffic directly or indirectly destined to a router is handled by the control plane. Management Plane Protection operates within the Control Plane Infrastructure.

# Management Plane

The *management plane* is the logical path of all traffic that is related to the management of a routing platform. One of three planes in a communication architecture that is structured in layers and planes, the management plane performs management functions for a network and coordinates functions among all the planes (management, control, and data). In addition, the management plane is used to manage a device through its connection to the network.

Examples of protocols processed in the management plane are Simple Network Management Protocol (SNMP), Telnet, HTTP, Secure HTTP (HTTPS), and SSH. These management protocols are used for monitoring and for command-line interface (CLI) access. Restricting access to devices to internal sources (trusted networks) is critical.

# Management Plane Protection Feature

The MPP protection feature, as well as all the management protocols under MPP, are disabled by default. When you configure an interface as either out-of-band or inband, it automatically enables MPP. Consequently, this enablement extends to all the protocols under MPP.

If MPP is disabled and a protocol is activated, all interfaces can pass traffic.

When MPP is enabled with an activated protocol, the only default management interfaces allowing management traffic are the route processor (RP) and standby route processor (SRP) Ethernet interfaces. You must manually configure any other interface for which you want to enable MPP as a management interface, using the MPP CLI that follows. Afterwards, only the default management interfaces and those you have previously configured as MPP interfaces will accept network management packets destined for the device. All other interfaces drop such packets.

**Note** Logical interfaces (or any other interfaces not present on the data plane) filter packets based on the ingress physical interface.

After configuration, you can modify or delete a management interface.

Following are the management protocols that the MPP feature supports. These management protocols are also the only protocols affected when MPP is enabled.

- SSH, v1 and v2
- SNMP, all versions
- Telnet
- TFTP
- HTTP
- HTTPS

## Benefits of the Management Plane Protection Feature

Implementing the MPP feature provides the following benefits:

- Greater access control for managing a device than allowing management protocols on all interfaces.
- Improved performance for data packets on non-management interfaces.
- Support for network scalability.
- Simplifies the task of using per-interface access control lists (ACLs) to restrict management access to the device.
- Fewer ACLs are needed to restrict access to the device.
- Prevention of packet floods on switching and routing interfaces from reaching the CPU.

# How to Configure a Device for Management Plane Protection

This section contains the following tasks:

## Configuring a Device for Management Plane Protection for an Inband Interface

Perform this task to configure a device that you have just added to your network or a device already operating in your network. This task shows how to configure MPP as an inband interface in which Telnet is allowed to access the router only through a specific interface.

Perform the following additional tasks to configure an inband MPP interface in non-default VRF.

- Configure the interface under the non-default inband VRF.
- Configure the global inband VRF.

- In the case of Telnet, configure the Telnet VRF server for the inband VRF.

## SUMMARY STEPS

1. **configure**
2. control-plane
3. management-plane
4. inband
5. **interface** {*type instance* | **all**}
6. **allow** {*protocol* | **all**} [**peer**]
7. **address ipv4** {*peer-ip-address* | *peer ip-address/length*}
8. Use the **commit** or **end** command.
9. **show mgmt-plane** [**inband** | **out-of-band**] [**interface** {*type instance*}]

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | control-plane<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# control-plane`<br>`RP/0/RP0/CPU0:router(config-ctrl)#` | Enters control plane configuration mode. |
| **Step 3** | management-plane<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ctrl)# management-plane`<br>`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<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-mpp)# inband`<br>`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**}<br>**Example:** | Configures a specific inband interface, or all inband interfaces. Use the **interface** command to enter management plane protection inband interface configuration mode.<br>• Use the **all** keyword to configure all interfaces. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-mpp-inband)# interface GigabitEthernet 0/6/0/1`<br>`RP/0/RP0/CPU0:router(config-mpp-inband-Gi0_6_0_1)#` | |
| **Step 6** | **allow** {*protocol* \| **all**} [**peer**]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-mpp-inband-Gi0_6_0_1)# allow Telnet peer`<br>`RP/0/RP0/CPU0:router(config-telnet-peer)#` | Configures an interface as an inband interface for a specified protocol or all protocols.<br>• Use the *protocol* argument to allow management protocols on the designated management interface.<br>  • HTTP or HTTPS<br>  • SNMP (also versions)<br>  • Secure Shell (v1 and v2)<br>  • TFTP<br>  • Telnet<br>• Use the **all** keyword to configure the interface to allow all the management traffic that is specified in the list of protocols.<br>• (Optional) Use the **peer** keyword to configure the peer address on the interface. |
| **Step 7** | **address ipv4** {*peer-ip-address* \| *peer ip-address/length*}<br>**Example:**<br>`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.<br>• Use the *peer-ip-address* argument to configure the peer IPv4 address in which management traffic is allowed on the interface.<br>• Use the *peer ip-address/length* argument to configure the prefix of the peer IPv4 address. |
| **Step 8** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 9** | **show mgmt-plane** [**inband** \| **out-of-band**] [**interface** {*type instance*}] | Displays information about the management plane, such as type of interface and protocols enabled on the interface. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br><br>`RP/0/RP0/CPU0:router# show mgmt-plane inband interface GigabitEthernet 0/6/0/1` | • (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.<br>• (Optional) Use the **out-of-band** keyword to display the out-of-band interface configurations.<br>• (Optional) Use the **interface** keyword to display the details for a specific interface. |

# Configuring a Device for Management Plane Protection for an Out-of-band Interface

Perform the following tasks to configure an out-of-band MPP interface.

- Configure the interface under the out-of-band VRF.
- Configure the global out-of-band VRF.
- In the case of Telnet, configure the Telnet VRF server for the out-of-band VRF.

### SUMMARY STEPS

1. **configure**
2. control-plane
3. management-plane
4. out-of-band
5. **vrf** *vrf-name*
6. **interface** {*type instance* | **all**}
7. **allow** {*protocol* | **all**} [**peer**]
8. **address ipv6** {*peer-ip-address* | *peer ip-address/length*}
9. Use the **commit** or **end** command.
10. **show mgmt-plane** [**inband** | **out-of-band**] [**interface** {*type instance*} | **vrf**]

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br><br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | control-plane<br>**Example:** | Enters control plane configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config)# control-plane`<br>`RP/0/RP0/CPU0:router(config-ctrl)#` | |
| **Step 3** | management-plane<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ctrl)#`<br>`management-plane`<br>`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<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-mpp)# out-of-band`<br>`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*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-mpp-outband)# vrf target` | Configures a Virtual Private Network (VPN) routing and forwarding (VRF) reference of an out-of-band interface.<br>• Use the *vrf-name* argument to assign a name to a VRF. |
| **Step 6** | **interface** {*type instance* \| **all**}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-mpp-outband)#`<br>`interface GigabitEthernet 0/6/0/2`<br>`RP/0/RP0/CPU0:router(config-mpp-outband-Gi0_6_0_2)#` | Configures a specific out-of-band interface, or all out-of-band interfaces, as an out-of-band interface. Use the **interface** command to enter management plane protection out-of-band configuration mode.<br>• Use the **all** keyword to configure all interfaces. |
| **Step 7** | **allow** {*protocol* \| **all**} [**peer**]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-mpp-outband-Gi0_6_0_2)#`<br>` allow TFTP peer`<br>`RP/0/RP0/CPU0:router(config-tftp-peer)#` | Configures an interface as an out-of-band interface for a specified protocol or all protocols.<br>• Use the *protocol* argument to allow management protocols on the designated management interface.<br>  • HTTP or HTTPS<br>  • SNMP (also versions)<br>  • Secure Shell (v1 and v2)<br>  • TFTP |

| | Command or Action | Purpose |
|---|---|---|
| | | • Telnet<br>• Use the **all** keyword to configure the interface to allow all the management traffic that is specified in the list of protocols.<br>• (Optional) Use the **peer** keyword to configure the peer address on the interface. |
| **Step 8** | **address ipv6** {*peer-ip-address* \| *peer ip-address/length*}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-tftp-peer)# address ipv6 33::33` | Configures the peer IPv6 address in which management traffic is allowed on the interface.<br>• Use the *peer-ip-address* argument to configure the peer IPv6 address in which management traffic is allowed on the interface.<br>• Use the *peer ip-address/length* argument to configure the prefix of the peer IPv6 address. |
| **Step 9** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 10** | **show mgmt-plane** [**inband** \| **out-of-band**] [**interface** {*type instance*} \| **vrf**]<br>**Example:**<br>`RP/0/RP0/CPU0:router# show mgmt-plane out-of-band interface GigabitEthernet 0/6/0/2` | Displays information about the management plane, such as type of interface and protocols enabled on the interface.<br>• (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.<br>• (Optional) Use the **out-of-band** keyword to display the out-of-band interface configurations.<br>• (Optional) Use the **interface** keyword to display the details for a specific interface.<br>• (Optional) Use the **vrf** keyword to display the Virtual Private Network (VPN) routing and forwarding reference of an out-of-band interface. |

# Configuration Examples for Implementing Management Plane Protection

This section provides the following configuration example:

## Configuring Management Plane Protection: Example

The following example shows how to configure inband and out-of-band interfaces for a specific IP address under MPP:

```
configure
 control-plane
  management-plane
   inband
    interface all
     allow SSH
     !
    interface GigabitEthernet 0/6/0/0
     allow all
     allow SSH
     allow Telnet peer
      address ipv4 10.1.0.0/16
    !
   !
    interface GigabitEthernet 0/6/0/1
     allow Telnet peer
      address ipv4 10.1.0.0/16
     !
    !
   !
   out-of-band
    vrf my_out_of_band
    interface GigabitEthernet 0/6/0/2
     allow TFTP peer
      address ipv6 33::33
     !
    !
   !
  !
 !

show mgmt-plane

Management Plane Protection

inband interfaces
----------------------

interface - GigabitEthernet0_6_0_0
        ssh configured -
                All peers allowed
        telnet configured -
                peer v4 allowed - 10.1.0.0/16
        all configured -
                All peers allowed
interface - GigabitEthernet0_6_0_1
```

```
        telnet configured -
                peer v4 allowed - 10.1.0.0/16

interface - all
        all configured -
                All peers allowed

outband interfaces
----------------------
interface - POS0_6_0_2
        tftp configured -
                peer v6 allowed - 33::33

show mgmt-plane out-of-band vrf

Management Plane Protection -
        out-of-band VRF - my_out_of_band
```

# Additional References

The following sections provide references related to implementing management plane protection.

### Related Documents

| Related Topic | Document Title |
|---|---|
| MPP commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *Management Plane Protection Commands on System Monitoring Command Reference for Cisco CRS Routers* |

### Standards

| Standards | Title |
|---|---|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

### MIBs

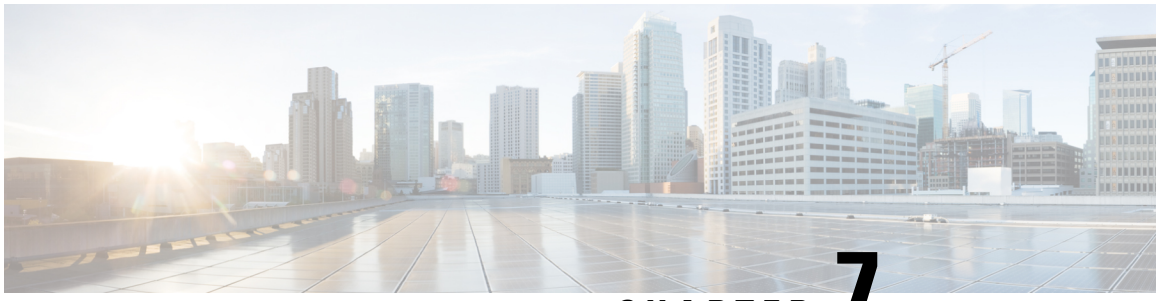
| MIBs | MIBs Link |
|---|---|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml |

### RFCs

| RFCs | Title |
|---|---|
| No new or modified RFCs are supported by this feature. | — |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 7

# Configuring Network Configuration Protocol

This module provides details of the Network Configuration Protocol. For relevant commands, see *System Security Command Reference for Cisco CRS Routers*.

| Release | Modification |
|---|---|
| Release 5.3.0 | This feature was introduced. |
| Release 5.3.1 | Support extended for more Yang models. |
| Release 6.0 | Support extended for the Netconf subsystem configuration to be vrf aware. The configuration of the netconf port is no longer sufficient to start the Netconf subsystem support. At least one vrf needs to be configured. The configuration of the port is now optional. |



# The Network Configuration Protocol

The Network Configuration Protocol (Netconf) provides mechanisms to install, manipulate, and delete the configuration of network devices. It uses an Extensible Markup Language (XML)-based data encoding for the configuration data as well as the protocol messages. Yang is a data modeling language used with Netconf.

Netconf uses a simple RPC-based (Remote Procedure Call) mechanism to facilitate communication between a client and a server. The client can be a script or application typically running as part of a network manager. The server is typically a network device.

The configuration of features need not be done the traditional way (using CLIs), the client application (controller) reads the Yang model and communicates with the Netconf server (IOS XR) accordingly.

**Note** Following are the deviations from IETF-NACM YANG, where the system does not support:

- The *ordered-by-user* functionality for rule-lists and rules. rule-lists & rules are sorted based on name.
- The *enable-nacm* leaf.
- The *notification* related leafs (notification-name & denied-notifications.)

# Netconf Sessions and Operations

A Netconf session is the logical connection between a network configuration application and a network device. A device should be capable of supporting multiple sessions and atleast one Netconf session.

Characteristics of a netconf session:

- Netconf is connection-oriented - SSH is the underlying transport.
- The netconf client establishes session with the server.
- Netconf sessions are established with the *hello* message. Features and capabilities are announced.
- Sessions can be terminated using the *close* or *kill* messages.

Basic Netconf operations:

- Get configuration <get-config>
- Get all information <get>
- Edit configuration <edit-config>
- Copy configuration <copy-config>

**Note** <copy-config> does not support source attribute with “data store” at present.

- <lock>, <unlock>
- <kill-session>
- <close-session>
- Commit configuration <commit>

# The Yang data model

Each feature has a defined Yang Model which is synthesized from the schemas. A model is published in a tree format and includes:

- Top level nodes and their subtrees

- Subtrees that augment nodes in other yang models

```
Example: The aaa Yang model
module: Cisco-IOS-XR-aaa-lib-cfg
   +--rw aaa
      +--rw accountings
      |  +--rw accounting* [type listname]
      |     +--rw type                xr:Cisco-ios-xr-string
      |     +--rw listname            xr:Cisco-ios-xr-string
      |     +--rw rp-failover?        Aaa-accounting-rp-failover
      |     +--rw broadcast?          Aaa-accounting-broadcast
      |     +--rw type-xr?            Aaa-accounting
      |     +--rw method*             Aaa-method
      |     +--rw server-group-name*   string
      +--rw authorizations
      |  +--rw authorization* [type listname]
      |     +--rw type                xr:Cisco-ios-xr-string
      |     +--rw listname            xr:Cisco-ios-xr-string
      |     +--rw method*             Aaa-method
      |     +--rw server-group-name*   string
      +--rw accounting-update!
      |  +--rw type                Aaa-accounting-update
      |  +--rw periodic-interval?   uint32
      +--rw authentications
         +--rw authentication* [type listname]
            +--rw type                xr:Cisco-ios-xr-string
            +--rw listname            xr:Cisco-ios-xr-string
            +--rw method*             Aaa-method
            +--rw server-group-name*   string
```

Advantages of using the Yang model are:

- Yang supports programmatic interfaces.
- Yang supports simplified network management applications.
- Yang supports interoperability that provides a standard way to model management data.

# Netconf and Yang

The workflow displayed here, will help the user to understand how Netconf-Yang can configure and control the network with minimal user intervention. The required components:

- Cisco Router (ASR9000 series or CRS) with Netconf capability
- Netconf Client Application with connection to the router

| S. No. | Device / component | Action |
|---|---|---|
| 1 | Cisco router (ASR 9000 or CRS router) | Login/ access the router. |
| 2 | Cisco router | Prerequisites for enabling Netconf.<br>• k9sec pie must be installed.<br>• Crypto keys must be generated. |

| S. No. | Device / component | Action |
|---|---|---|
| 3 | Cisco router | Enable Netconf agent. Use the **netconf-yang agent ssh** and **ssh server netconf** command. The port can be selected. By default, it is set as 830. |
| 4 | Cisco router | Yang models are a part of the software image. The models can be retrieved from the router , using the <get-schema> operation. |
| 5 | Netconf client (application)<br>The application can be on any standalone application or a SDN controller supporting Netconf | Installs and processes the Yang models.<br>The client can offer a list of supported yang models; else the user will have to browse and locate the required yang file.<br>There is a yang model file for each configuration module; for instance if the user wants to configure CDP , the relevant yang model is Cisco-IOS-XR-cdp-cfg<br>**Note** Refer the table which lists all the supported yang models. Supported Yang Models , on page 138 |
| 5 | Netconf client | Sends Netconf operation request over SSH to the router. A configuration request could include Yang-based XML data to the router. Currently, SSH is the only supported transport method. |
| 6 | Cisco router | Understands the Yang-based XML data and the network is configured accordingly (in case of configuration request from the client). |
| | | The interactions between the client and the router happens until the network is configured as desired. |

# Supported Yang Models

The Yang models can be downloaded from a prescribed location (ftp server) or can also be retrieved directly from the router using the get-schema operation.

For a feature, separate Yang models are available for configuring the feature and to get operational statistics (show commands). The **-cfg.yang** suffix denotes configuration and **-oper*.yang** is for operational data statistics. In some cases, **-oper** is followed by **-sub**, indicating that a submodule(s) is available.

For a list of supported Yang models, see https://github.com/YangModels/yang/tree/master/vendor/cisco/xr

# Denial of Services Defence for Netconf-Yang

In case of a DoS (Denial of Service) attack on Netconf, wherein, Netconf receives numerous requests in a short span of time, the router may become irresponsive if Netconf consumes most of the bandwidth or CPU

processing time. This can be prevented, by limiting the traffic directed at the Netconf agent. This is achieved using the **netconf-yang agent rate-limit** and **netconf-yang agent session** commands.

If rate-limit is set, the Netconf processor measures the incoming traffic from the SSH server. If the incoming traffic exceeds the set rate-limit, the packets are dropped.

If session-limit is set, the Netconf processor checks for the number of open sessions. If the number of current sessions is greater than or equal to, the set limit, no new sessions are opened.

Session idle- timeout and absolute-timeout also prevent DoS attacks. The Netconf processor closes the sessions, even without user input or intervention, as soon at the time out session is greater than or equal to the set time limit.

The relevant commands are discussed in detail, in the *System Security Command Reference for Cisco CRS Routers*

# Dynamic Loading of Operational Yang Models

Netconf is enhanced to pre-load only the configurational yang models in memory, when it starts. The operational yang models are loaded into memory only when a request is issued. This helps reduce consumption of the RAM memory.

# Enabling NETCONF over SSH

This task enables NETCONF over SSH. SSH is currently the only supported transport method .

If the client supports, Netconf over ssh can utilize the multi-channeling capabilities of IOS XR ssh server. For additional details about Multi-channeling in SSH, see *Implementing Secure Shell* in *System Security Configuration Guide*.

**Prerequisites:**

- k9sec pie must be installed, otherwise the port configuration for the netconf ssh server cannot be completed. (The Netconf subsystem for SSH, as well as, SSH cannot be configured without the k9sec pie.)
- Crypto keys must be generated prior to this configuration.
- The Netconf-YANG feature is packaged in the mgbl pie, which must be installed before enabling the Netconf-YANG agent.

## SUMMARY STEPS

1. **configure**
2. **netconf-yang agent ssh**
3. **ssh server netconf** [**vrf** *vrf-name* [**ipv4 access-list***ipv4 access list name*] [**ipv6 access-list** *ipv6 access list name*] ]
4. **ssh server netconf port** *port-number*

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **netconf-yang agent ssh**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # netconf agent ssh` | Enables NETCONF agent over SSH connection. After NETCONF is enabled, the Yang model in the controllcker, can configure the relevant models.<br>**Note** The Yang models can be retrieved from the router via NETCONF <get-schema> operation. |
| **Step 3** | **ssh server netconf** [**vrf** *vrf-name* [**ipv4 access-list***ipv4 access list name*] [**ipv6 access-list** *ipv6 access list name*]]<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # ssh server netconf vrf netconfvrf ipv4 access-list InternetFilter` | Brings up the netconf subsytem support with 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.<br>Optionally ACLs for IPv4 and IPv6 can be used to restrict access to the netconf subsystem of the ssh server before the port is opened.<br>**Note** The netconf subsystem support with SSH server can be configured for use with multiple VRFs . |
| **Step 4** | **ssh server netconf port** *port-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # ssh server netconf port 830` | Configures a port for the netconf ssh server. This command is optional. If no port is specified, port 830 is uses by default.<br>**Note** 830 is the IANA-assigned TCP port for NETCONF over SSH, but it can be changed using this command. |

### What to do next

The **show netconf-yang statistics** command and **show netconf-yang clients** command can be used to verify the configuration details of the netconf agent.

The **clear netconf-yang agent session** command clears the specified Netconf session (on the Netconf server side).

# Examples: Netconf over SSH

This section illustrates some examples relevant to Netconf:

**Enabling netconf-yang for ssh transport and netconf subsystem for default vrf with default port (830)**

```
config
netconf-yang agent ssh
```

```
ssh server netconf vrf default
  !
!
```

### Enabling netconf-yang for ssh transport and netconf subsystem for vrf *green* and vrf *red* with netconf port (831)

```
config
netconf-yang agent ssh
!
ssh server netconf vrf green
ssh server netconf vrf red
ssh server netconf port 831
  !
!
```

### Show command outputs

```
show netconf-yang statistics
Summary statistics        requests|             total time|   min time per request|   max
time per request|   avg time per request|
other                              0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
close-session                      4|       0h  0m  0s   3ms|       0h  0m  0s   0ms|
 0h  0m  0s   1ms|       0h  0m  0s   0ms|
kill-session                       0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
get-schema                         0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
get                                0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s
get-config                         1|       0h  0m  0s   1ms|       0h  0m  0s   1ms|
 0h  0m  0s   1ms|       0h  0m  0s   1ms|
edit-config                        3|       0h  0m  0s   2ms|       0h  0m  0s   0ms|
 0h  0m  0s   1ms|       0h  0m  0s   0ms|
commit                             0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
cancel-commit                      0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
lock                               0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
unlock                             0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
discard-changes                    0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
validate                           0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|

show netconf-yang clients
client session ID|   NC version|    client connect time|        last OP time|        last
OP type|    <lock>|
22969|                1.1|         0d  0h  0m  2s|             11:11:24|
close-session|        No|
15389|               1.1|        0d  0h  0m  1s|            11:11:25|         get-config|
          No|
```

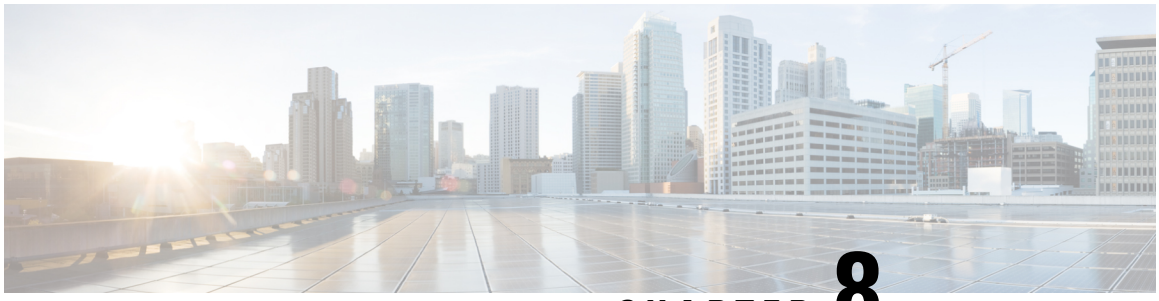
# Additional Reference

*Table 7: Related Documents*

| Related Topic | Document Title |
|---|---|
| Netconf-Yang | For related commands, see *System Security Command Reference for Cisco CRS Routers* |

*Table 8: Standards*

| Component | RFCs |
|---|---|
| YANG | 6020 |
| NETCONF | 6241 |
| NETCONF over SSH | 6242 |



CHAPTER 8

# Configuring Software Authentication Manager

Software Authentication Manager (SAM) is a component of the the Cisco IOS XR Software operating system that ensures that software being installed on the router is safe, and that the software does not run if its integrity has been compromised.

For information on SAM commands, see the *Software Authentication Manager Commands* module in *System Security Command Reference for Cisco CRS Routers*.

For information on setting the system clock, see the **clock set** command in *Clock Commands* module in *System Management Command Reference for Cisco CRS Routers* .

**Feature History for Configuring Software Authentication Manager**

| Release | Modification |
|---|---|
| Release 3.5.0 | This feature was introduced. |



# Prerequisites for Configuring Software Authentication Manager

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

# Information about Software Authentication Manager

For SAM to verify software during installation, the software to be installed must be in a Packager for IOS/ENA (PIE) format. PIEs are digitally signed and SAM verifies the digital signature before allowing bits from that PIE to reside on the router. Each time an installed piece of software is run, SAM ensures that the integrity of the software is not been compromised since it was installed. SAM also verifies that software preinstalled on a flash card has not been tampered with while in transit.

When the initial image or a software package update is loaded on the router, SAM verifies the validity of the image by checking the expiration date of the certificate used to sign the image. If an error message is displayed

indicating that your certificate has expired, check the system clock and verify that it is accurate. If the system clock is not set correctly, the system does not function properly.

# How to set up a Prompt Interval for the Software Authentication Manager

When the SAM detects an abnormal condition during boot time, it prompts the user to take action and waits for a certain interval. When the user does not respond within this interval, SAM proceeds with a predetermined action that can also be configured.

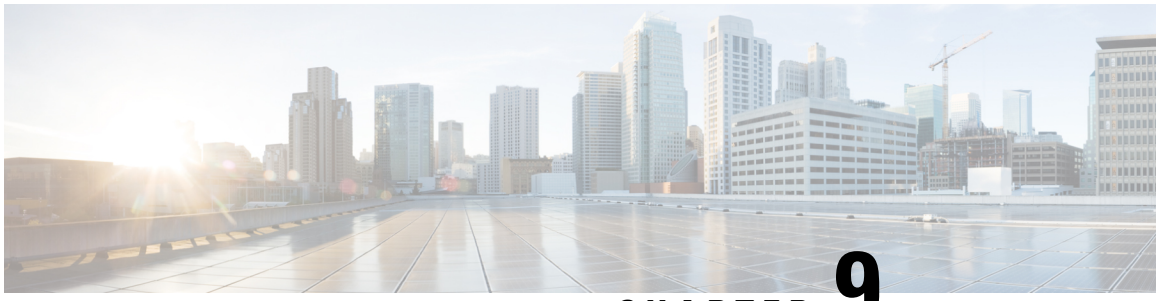
To set up the Prompt Interval, perform the following tasks.

### SUMMARY STEPS

1. **configure**
2. **sam promptinterval** *time-interval* {**proceed** | **terminate**}
3. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **sam promptinterval** *time-interval* {**proceed** | **terminate**}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# sam prompt-interval 25 {proceed \| terminate}` | Sets the prompt interval in seconds, after which the SAM either proceeds or terminates the interval. The Prompt interval ranges from 0 to 300 seconds.<br>If the user responds, SAM considers it as a 'Yes' and proceeds with the next action. If the user does not respond, SAM considers it as a 'No' and terminates the action. The default time for which SAM waits is 10 seconds. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |



CHAPTER 9

# Implementing Secure Shell

Secure Shell (SSH) is an application and a protocol that provides a secure replacement to the Berkeley r-tools. The protocol secures sessions using standard cryptographic mechanisms, and the application can be used similarly to the Berkeley **rexec** and **rsh** tools.

Two versions of the SSH server are available: SSH Version 1 (SSHv1) and SSH Version 2 (SSHv2). SSHv1 uses Rivest, Shamir, and Adelman (RSA) keys and SSHv2 uses either Digital Signature Algorithm (DSA) keys or Rivest, Shamir, and Adelman (RSA) keys, or Elliptic Curve Digital Signature Algorithm (ECDSA) keys. Cisco IOS XR software supports both SSHv1 and SSHv2.

**Note** Cisco IOS XR does not support X11 forwarding through an SSH connection.

This module describes how to implement Secure Shell on the the Cisco IOS XR Software.

**Note** For a complete description of the Secure Shell commands used in this chapter, see the *Secure Shell Commands* module in *System Security Command Reference for Cisco CRS Routers*.

### Feature History for Implementing Secure Shell

| Release | Modification |
|---|---|
| Release 2.0 | This feature was introduced. |
| Release 3.8.0 | • Support for Advanced Encryption Standard (AES) algorithm was added.<br>• The SSH server and client were made VRF-aware. |
| Release 3.9.0 | Support was added for the following enhancements:<br>• RSA based authentication on the SSH server<br>• SFTP client in interactive mode<br>• SFTP server implementation |
| Release 5.1.1 | Support was added for SSH Multiplexing |

| Release | Modification |
|---|---|
| Release 6.0 | Support extended for the Netconf subsystem configuration to be vrf aware. At least one vrf needs to be configured to start the Netconf subsystem support. The configuration of the port is now optional. |
| Release 6.4.1 | Support was added for ECDSA algorithm on IOS-XR SSHv2. |



# Prerequisites for Implementing Secure Shell

The following prerequisites are required to implement Secure Shell:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- Download the required image on your router. The SSH server and SSH client require you to have a a crypto package (data encryption standard [DES], 3DES and AES) from Cisco downloaded on your router.
- To run an SSHv2 server, you must have a VRF. This may be the default VRF or a specific VRF. VRF changes are applicable only to the SSH v2 server.
- Configure user authentication for local or remote access. You can configure authentication with or without authentication, authorization, and accounting (AAA). For more information, see the *Authentication, Authorization, and Accounting Commands on* Cisco IOS XR Software module in the *System Security Command Reference for Cisco CRS Routers* publication and *Configuring AAA Services on* Cisco IOS XR Software module in the *System Security Configuration Guide for Cisco CRS Routers* publication.
- 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:

- A VRF is not accepted as inband if that VRF is already set as an out-of-band VRF. SSH v1 continues to bind only to the default VRF.

- In order for an outside client to connect to the router, the router needs to have an RSA (for SSHv1 or SSHv2) or DSA (for SSHv2) or ECDSA (for SSHv2) key pair configured. ECDSA, DSA and RSA keys are not required if you are initiating an SSH client connection from the router to an outside routing device. The same is true for SFTP: ECDSA, DSA and RSA keys are not required because SFTP operates only in client mode.
- In order for SFTP to work properly, the remote SSH server must enable the SFTP server functionality. For example, the SSHv2 server is configured to handle the SFTP subsystem with a line such as **/etc/ssh2/sshd2_config**:
- **subsystem-sftp /usr/local/sbin/sftp-server**
- The SFTP server is usually included as part of SSH packages from public domain and is turned on by default configuration.
- SFTP is compatible with sftp server version OpenSSH_2.9.9p2 or higher.
- RSA-based user authentication is supported in the SSH and SFTP servers. The support however, is not extended to the SSH client.
- Execution shell and SFTP are the only applications supported.
- The AES encryption algorithm is supported on the SSHv2 server and client, but not on the SSHv1 server and client. Any requests for an AES cipher sent by an SSHv2 client to an SSHv1 server are ignored, with the server using 3DES instead.
- The SFTP client does not support remote filenames containing wildcards (* ,?, []). The user must issue the **sftp** command multiple times or list all of the source files from the remote host to download them on to the router. For uploading, the router SFTP client can support multiple files specified using a wildcard provided that the issues mentioned in the first through third bullets in this section are resolved.
- The cipher preference for the SSH server follows the order AES128, AES192, AES256, and, finally, 3DES. The server rejects any requests by the client for an unsupported cipher, and the SSH session does not proceed.
- Use of a terminal type other than vt100 is unsupported, and the software generates a warning message in this case.
- Password messages of “none” are unsupported on the SSH client.
- Because the router infrastructure does not provide support for UNIX-like file permissions, files created on the local device lose the original permission information. For files created on the remote file system, the file permission adheres to the umask on the destination host and the modification and last access times are the time of the copy.

# Information About Implementing Secure Shell

To implement SSH, you should understand the following concepts:

## SSH Server

The SSH server feature enables an SSH client to make a secure, encrypted connection to a Cisco router. This connection provides functionality that is similar to that of an inbound Telnet connection. Before SSH, security

was limited to Telnet security. SSH allows a strong encryption to be used with the Cisco IOS XR software authentication. The SSH server in Cisco IOS XR software works with publicly and commercially available SSH clients.

# SSH Client

The SSH client feature is an application running over the SSH protocol to provide device authentication and encryption. The SSH client enables a Cisco router to make a secure, encrypted connection to another Cisco router or to any other device running the SSH server. This connection provides functionality that is similar to that of an outbound Telnet connection except that the connection is encrypted. With authentication and encryption, the SSH client allows for a secure communication over an insecure network.

The SSH client in the Cisco IOS XR software worked with publicly and commercially available SSH servers. The SSH client supported the ciphers of AES, 3DES, message digest algorithm 5 (MD5), SHA1, and password authentication. User authentication was performed in the Telnet session to the router. The user authentication mechanisms supported for SSH were RADIUS, TACACS+, and the use of locally stored usernames and passwords.

The SSH client supports setting DSCP value in the outgoing packets.

```
ssh client dscp <value from 0 - 63>
```

If not configured, the default DSCP value set in packets is 16 (for both client and server).

The SSH client supports the following options:

- DSCP—DSCP value for SSH client sessions.

```
RP/0/5/CPU0:router#configure
RP/0/5/CPU0:router(config)#ssh ?
  client   Provide SSH client service
  server   Provide SSH server service
  timeout  Set timeout value for SSH
RP/0/5/CPU0:router(config)#ssh client ?
```

- Knownhost—Enable the host pubkey check by local database.
- Source-interface—Source interface for SSH client sessions.

```
RP/0/5/CPU0:router(config)#ssh client source-interface ?
  ATM                 ATM Network Interface(s)
  BVI                 Bridge-Group Virtual Interface
  Bundle-Ether        Aggregated Ethernet interface(s)
  Bundle-POS          Aggregated POS interface(s)
  CEM                 Circuit Emulation interface(s)
  GigabitEthernet     GigabitEthernet/IEEE 802.3 interface(s)
  IMA                 ATM Network Interface(s)
  IMtestmain          IM Test Interface
  Loopback            Loopback interface(s)
  MgmtEth             Ethernet/IEEE 802.3 interface(s)
  Multilink           Multilink network interface(s)
  Null                Null interface
  PFItestmain         PFI Test Interface
  PFItestnothw        PFI Test Not-HW Interface
  POS                 Packet over SONET/SDH network interface(s)
  PW-Ether            PWHE Ethernet Interface
  PW-IW               PWHE VC11 IP Interworking Interface
  Serial              Serial network interface(s)
  VASILeft            VASI Left interface(s)
  VASIRight           VASI Right interface(s)
  test-bundle-channel  Aggregated Test Bundle interface(s)
  tunnel-ipsec        IPSec Tunnel interface(s)
```

```
    tunnel-mte           MPLS Traffic Engineering P2MP Tunnel interface(s)
    tunnel-te            MPLS Traffic Engineering Tunnel interface(s)
    tunnel-tp            MPLS Transport Protocol Tunnel interface
  RP/0/5/CPU0:router(config)#ssh client source-interface
  RP/0/5/CPU0:router(config)#
```

- VRF—Source interface VRF for SSH client sessions:

```
  RP/0/5/CPU0:router(config)#ssh client vrf ?
    WORD  VRF name (max:32 chars)
  RP/0/5/CPU0:router(config)#ssh client vrf shan ?
    <cr>
  RP/0/5/CPU0:router(config)#ssh client vrf shan
```

SSH also supports remote command execution as follows:

```
RP/0/5/CPU0:router#ssh ?
  A.B.C.D  IPv4 (A.B.C.D) address
  WORD     Hostname of the remote node
  X:X::X   IPv6 (A:B:C:D...:D) address
  vrf      vrf table for the route lookup
RP/0/5/CPU0:router#ssh 10.1.1.1 ?
  cipher            Accept cipher type
  command           Specify remote command (non-interactive)
  source-interface  Specify source interface
  username          Accept userid for authentication
  <cr>
RP/0/5/CPU0:router#ssh 192.68.46.6 username admin command "show redundancy sum"
Password:


Wed Jan  9 07:05:27.997 PST
    Active Node    Standby Node
    -----------    ------------
       0/4/CPU0        0/5/CPU0 (Node Ready, NSR: Not Configured)

RP/0/5/CPU0:router#
```

# SFTP Feature Overview

SSH includes support for standard file transfer protocol (SFTP) , a new standard file transfer protocol introduced in SSHv2. This feature provides a secure and authenticated method for copying router configuration or router image files.

The SFTP client functionality is provided as part of the SSH component and is always enabled on the router. Therefore, a user with the appropriate level can copy files to and from the router. Like the **copy** command, the **sftp** command can be used only in EXEC mode.

The SFTP client is VRF-aware, and you may configure the secure FTP client to use the VRF associated with a particular source interface during connections attempts. The SFTP client also supports interactive mode, where the user can log on to the server to perform specific tasks via the Unix server.

The SFTP Server is a sub-system of the SSH server. In other words, when an SSH server receives an SFTP server request, the SFTP API creates the SFTP server as a child process to the SSH server. A new SFTP server instance is created with each new request.

The SFTP requests for a new SFTP server in the following steps:

- The user runs the **sftp** command with the required arguments
- The SFTP API internally creates a child session that interacts with the SSH server

- The SSH server creates the SFTP server child process
- The SFTP server and client interact with each other in an encrypted format
- The SFTP transfer is subject to LPTS policer "SSH-Known". Low policer values will affect SFTP transfer speeds

**Note** In IOS-XR SW release 4.3.1 onwards the default policer value for SSH-Known has been reset from 2500pps to 300pps. Slower transfers are expected due to this change. You can adjust the lpts policer value for this punt cause to higher values that will allow faster transfers

When the SSH server establishes a new connection with the SSH client, the server daemon creates a new SSH server child process. The child server process builds a secure communications channel between the SSH client and server via key exchange and user authentication processes. If the SSH server receives a request for the sub-system to be an SFTP server, the SSH server daemon creates the SFTP server child process. For each incoming SFTP server subsystem request, a new SSH server child and a SFTP server instance is created. The SFTP server authenticates the user session and initiates a connection. It sets the environment for the client and the default directory for the user.

Once the initialization occurs, the SFTP server waits for the SSH_FXP_INIT message from the client, which is essential to start the file communication session. This message may then be followed by any message based on the client request. Here, the protocol adopts a 'request-response' model, where the client sends a request to the server; the server processes this request and sends a response.

The SFTP server displays the following responses:

- Status Response
- Handle Response
- Data Response
- Name Response

**Note** The server must be running in order to accept incoming SFTP connections.

# RSA Based Host Authentication

Verifying the authenticity of a server is the first step to a secure SSH connection. This process is called the host authentication, and is conducted to ensure that a client connects to a valid server.

The host authentication is performed using the public key of a server. The server, during the key-exchange phase, provides its public key to the client. The client checks its database for known hosts of this server and the corresponding public-key. If the client fails to find the server's IP address, it displays a warning message to the user, offering an option to either save the public key or discard it. If the server’s IP address is found, but the public-key does not match, the client closes the connection. If the public key is valid, the server is verified and a secure SSH connection is established.

The IOS XR SSH server and client had support for DSA based host authentication. But for compatibility with other products, like IOS, RSA based host authentication support is also added.

# RSA Based User Authentication

One of the method for authenticating the user in SSH protocol is RSA public-key based user authentication. The possession of a private key serves as the authentication of the user. This method works by sending a signature created with a private key of the user. Each user has a RSA keypair on the client machine. The private key of the RSA keypair remains on the client machine.

The user generates an RSA public-private key pair on a unix client using a standard key generation mechanism such as ssh-keygen. The max length of the keys supported is 4096 bits, and the minimum length is 512 bits. The following example displays a typical key generation activity:

```
bash-2.05b$ ssh-keygen –b 1024 –t rsa
Generating RSA private key, 1024 bit long modulus
```

The public key must be in base64 encoded (binary) format for it to be imported correctly into the box. You can use third party tools available on the Internet to convert the key to the binary format.

Once the public key is imported to the router, the SSH client can choose to use the public key authentication method by specifying the request using the "-o" option in the SSH client. For example:

```
client$ ssh -o PreferredAuthentications=publickey 1.2.3.4
```

If a public key is not imported to a router using the RSA method, the SSH server initiates the password based authentication. If a public key is imported, the server proposes the use of both the methods. The SSH client then chooses to use either method to establish the connection. The system allows only 10 outgoing SSH client connections.

Currently, only SSH version 2 and SFTP server support the RSA based authentication. For more information on how to import the public key to the router, see the *Implementing Certification Authority Interoperability on the Cisco IOS XR Software* chapter in this guide.

**Note** The preferred method of authentication would be as stated in the SSH RFC. The RSA based authentication support is only for local authentication, and not for TACACS/RADIUS servers.

Authentication, Authorization, and Accounting (AAA) is a suite of network security services that provide the primary framework through which access control can be set up on your Cisco router or access server. For more information on AAA, see the *Authentication, Authorization, and Accounting Commands on* the Cisco IOS XR Software module in the *System Security Command Reference for Cisco CRS Routers* publication and the *Configuring AAA Services on* the Cisco IOS XR Software *Software*module in the *System Security Configuration Guide for Cisco CRS Routers* publication.

# SSHv2 Client Keyboard-Interactive Authentication

An authentication method in which the authentication information is entered using a keyboard is known as keyboard-interactive authentication. This method is an interactive authentication method in the SSH protocol. This type of authentication allows the SSH client to support different methods of authentication without having to be aware of their underlying mechanisms.

Currently, the SSHv2 client supports the keyboard-interactive authentication. This type of authentication works only for interactive applications.

**Note** The password authentication is the default authentication method. The keyboard-interactive authentication method is selected if the server is configured to support only the keyboard-interactive authentication.

# The Network Configuration Protocol

The Network Configuration Protocol (Netconf) provides mechanisms to install, manipulate, and delete the configuration of network devices. It uses an Extensible Markup Language (XML)-based data encoding for the configuration data as well as the protocol messages. Yang is a data modeling language used with Netconf.

Netconf uses a simple RPC-based (Remote Procedure Call) mechanism to facilitate communication between a client and a server. The client can be a script or application typically running as part of a network manager. The server is typically a network device.

The configuration of features need not be done the traditional way (using CLIs), the client application (controller) reads the Yang model and communicates with the Netconf server (IOS XR) accordingly.

**Note** Following are the deviations from IETF-NACM YANG, where the system does not support:

- The *ordered-by-user* functionality for rule-lists and rules. rule-lists & rules are sorted based on name.
- The *enable-nacm* leaf.
- The *notification* related leafs (notification-name & denied-notifications.)

## Netconf Sessions and Operations

A Netconf session is the logical connection between a network configuration application and a network device. A device should be capable of supporting multiple sessions and atleast one Netconf session.

Characteristics of a netconf session:

- Netconf is connection-oriented - SSH is the underlying transport.
- The netconf client establishes session with the server.
- Netconf sessions are established with the *hello* message. Features and capabilities are announced.
- Sessions can be terminated using the *close* or *kill* messages.

Basic Netconf operations:

- Get configuration <get-config>
- Get all information <get>
- Edit configuration <edit-config>
- Copy configuration <copy-config>

**Note** <copy-config> does not support source attribute with “data store” at present.

- <lock>, <unlock>
- <kill-session>
- <close-session>
- Commit configuration <commit>

# The Yang data model

Each feature has a defined Yang Model which is synthesized from the schemas. A model is published in a tree format and includes:

- Top level nodes and their subtrees
- Subtrees that augment nodes in other yang models

```
Example: The aaa Yang model
module: Cisco-IOS-XR-aaa-lib-cfg
   +--rw aaa
      +--rw accountings
      |  +--rw accounting* [type listname]
      |     +--rw type                 xr:Cisco-ios-xr-string
      |     +--rw listname             xr:Cisco-ios-xr-string
      |     +--rw rp-failover?         Aaa-accounting-rp-failover
      |     +--rw broadcast?           Aaa-accounting-broadcast
      |     +--rw type-xr?             Aaa-accounting
      |     +--rw method*              Aaa-method
      |     +--rw server-group-name*   string
      +--rw authorizations
      |  +--rw authorization* [type listname]
      |     +--rw type                 xr:Cisco-ios-xr-string
      |     +--rw listname             xr:Cisco-ios-xr-string
      |     +--rw method*              Aaa-method
      |     +--rw server-group-name*   string
      +--rw accounting-update!
      |  +--rw type                 Aaa-accounting-update
      |  +--rw periodic-interval?   uint32
      +--rw authentications
         +--rw authentication* [type listname]
            +--rw type                 xr:Cisco-ios-xr-string
            +--rw listname             xr:Cisco-ios-xr-string
            +--rw method*              Aaa-method
            +--rw server-group-name*   string
```

Advantages of using the Yang model are:

- Yang supports programmatic interfaces.
- Yang supports simplified network management applications.
- Yang supports interoperability that provides a standard way to model management data.

# How to Implement Secure Shell

To configure SSH, perform the tasks described in the following sections:

## Configuring SSH

**Note** For SSHv1 configuration, Step 1 to Step 4 are required. For SSHv2 configuration, Step 2 to Step 4 are optional.

SSH server supports setting DSCP value in the outgoing packets.

```
ssh server dscp <value from 0 - 63>
```

If not configured, the default DSCP value set in packets is 16 (for both client and server).

This is the syntax for setting DSCP value:

```
RP/0/5/CPU0:router(config)#ssh server dscp ?
  <0-63>  DSCP value range

RP/0/5/CPU0:router(config)#ssh server dscp 63 ?
  <cr>
RP/0/5/CPU0:router(config)#ssh server dscp 63
RP/0/5/CPU0:router(config)#

RP/0/5/CPU0:router(config)#ssh client dscp ?
  <0-63>  DSCP value range

RP/0/5/CPU0:router(config)#ssh client dscp 0 ?
  <cr>
RP/0/5/CPU0:router(config)#ssh client dscp 0
RP/0/5/CPU0:router(config)#
```

Perform this task to configure SSH.

### SUMMARY STEPS

1. **configure**
2. **hostname** *hostname*
3. **domain name** *domain-name*
4. Use the **commit** or **end** command.
5. **crypto key generate rsa** [**usage keys** | **general-keys**] [*keypair-label*]
6. **crypto key generate dsa**
7. **crypto key generate ecdsa** [**nistp256** | **nistp384** | **nistp521**]
8. **configure**
9. **ssh timeout** *seconds*
10. Do one of the following:
    - **ssh server** [**vrf** *vrf-name* [**ipv4 access-list**IPv4 access-list name] [**ipv6 access-list** IPv6 access-list name]]
    - **ssh server v2**

11. Use the **commit** or **end** command.
12. **show ssh**
13. **show ssh session details**
14. **show ssh history**
15. **show ssh history details**
16. **show tech-support ssh**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **hostname** *hostname*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# hostname router1` | Configures a hostname for your router. |
| **Step 3** | **domain name** *domain-name*<br>**Example:**<br>`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** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 5** | **crypto key generate rsa** [**usage keys** \| **general-keys**] [*keypair-label*]<br>**Example:**<br>`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.<br>• To delete the RSA key pair, use the **crypto key zeroize rsa** command.<br>• This command is used for SSHv1 only. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **crypto key generate dsa**<br>**Example:**<br>`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.<br>• The recommended minimum modulus size is 1024 bits.<br>• Generates a DSA key pair.<br>To delete the DSA key pair, use the **crypto key zeroize dsa** command.<br>• This command is used only for SSHv2. |
| **Step 7** | **crypto key generate ecdsa** [**nistp256** \| **nistp384** \| **nistp521**]<br>**Example:**<br>`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.<br>• To delete the ECDSA key pair, use the **crypto key zeroize ecdsa [ nistp256 \| nistp384 \| nistp521]** command.<br>• This command is used for SSHv2 only. |
| **Step 8** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 9** | **ssh timeout** *seconds*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# ssh timeout 60` | (Optional) Configures the timeout value for user authentication to AAA.<br>• If the user fails to authenticate itself to AAA within the configured time, the connection is terminated.<br>• 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:<br>• **ssh server** [**vrf** *vrf-name* [**ipv4 access-list**IPv4 access-list name] [**ipv6 access-list** IPv6 access-list name]]<br>• **ssh server v2**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# ssh server vrf green ipv4 access-list list1 ipv6 access-list list 2`<br>or<br>`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.Optionally ACLs for IPv4 and IPv6 can be used to restrict access to the server before the port is opened. To stop the SSH server from receiving any further connections for the specified VRF, use the **no** form of this command. If no VRF is specified, the default is assumed.<br>**Note** The SSH server can be configured for multiple VRF usage. |

| | Command or Action | Purpose |
|---|---|---|
| | | • (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** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 12** | **show ssh**<br>**Example:**<br>`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**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show ssh session details` | (Optional) Displays a detailed report of the SSHv2 connections to and from the router. |
| **Step 14** | **show ssh history**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show ssh history` | (Optional) Displays the last hundred SSH connections that were terminated. |
| **Step 15** | **show ssh history details**<br>**Example:**<br>`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**<br>**Example:**<br>`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

# Automatic Generation of SSH Host-Key Pairs

This feature brings in the functionality of automatically generating the SSH host-key pairs for the DSA, ECDSA (such as **ecdsa-nistp256**, **ecdsa-nistp384**, and **ecdsa-nistp521**) and RSA algorithms. This in turn eliminates the need for explicitly generating each SSH host-key pair after the router boots up. Because the keys are already present in the system, the SSH client can establish connection with the SSH server soon after the router boots up with the basic SSH configuration. This is useful especially during zero touch provisioning (ZTP) and Golden ISO boot up scenarios.

Before the introduction of this feature, you had to execute the **crypto key generate** command in EXEC mode to generate the required SSH host-key pairs.

Although the host-key pairs are auto-generated with the introduction of this feature, you still have the flexibility to select only the required algorithms on the SSH server. You can use the **ssh server algorithms host-key** command in Global Configuration mode to achieve the same. Alternatively, you can also use the existing **crypto key zeroize** command in EXEC mode to remove the algorithms that are not required.

**Note** In a system upgrade scenario from version 1 to version 2, the system does not generate the SSH host-key pairs automatically if they were already generated in version 1. The host-key pairs are generated automatically only if they were not generated in version 1.

## Configure the Allowed SSH Host-Key Pair Algorithms

When the SSH client attempts a connection with the SSH server, it sends a list of SSH host-key pair algorithms (in the order of preference) internally in the connection request. The SSH server, in turn, picks the first matching algorithm from this request list. The server establishes a connection only if that host-key pair is already generated in the system, and if it is configured (using the **ssh server algorithms host-key** command) as the allowed algorithm.

**Note** If this configuration of allowed host-key pairs is not present in the SSH server, then you can consider that the SSH server allows all host-key pairs. In that case, the SSH client can connect with any one of the host-key pairs. Not having this configuration also ensures backward compatibility in system upgrade scenarios.

### Configuration Example

You may perform this (optional) task to specify the allowed SSH host-key pair algorithm (in this example, **ecdsa**) from the list of auto-generated host-key pairs on the SSH server:

```
/* Example to select the ecdsa algorithm */
Router(config)#ssh server algorithms host-key ecdsa-nistp521
```

Similarly, you may configure other algorithms.

### Running Configuration

```
ssh server algorithms host-key ecdsa-nistp521
!
```

### Verify the SSH Host-Key Pair Algorithms

**Note** With the introduction of the automatic generation of SSH host-key pairs, the output of the **show crypto key mypubkey** command displays key information of all the keys that are auto-generated. Before its introduction, the output of this show command displayed key information of only those keys that you explicitly generated using the **crypto key generate** command.

```
Router#show crypto key mypubkey ecdsa
Mon Nov 19 12:22:51.762 UTC
Key label: the_default
Type     : ECDSA General Curve Nistp256
Degree   : 256
Created  : 10:59:08 UTC Mon Nov 19 2018
Data     :
04AC7533 3ABE7874 43F024C1 9C24CC66 490E83BE 76CEF4E2 51BBEF11 170CDB26
14289D03 6625FC4F 3E7F8F45 0DA730C3 31E960FE CF511A05 2B0AA63E 9C022482
6E

Key label: the_default
Type     : ECDSA General Curve Nistp384
Degree   : 384
Created  : 10:59:08 UTC Mon Nov 19 2018
Data     :
04B70BAF C096E2CA D848EE72 6562F3CC 9F12FA40 BE09BFE6 AF0CA179 F29F6407
FEE24A43 84C5A5DE D7912208 CB67EE41 58CB9640 05E9421F 2DCDC41C EED31288
6CACC8DD 861DC887 98E535C4 893CB19F 5ED3F6BC 2C90C39B 10EAED57 87E96F78
B6

Key label: the_default
Type     : ECDSA General Curve Nistp521
Degree   : 521
Created  : 10:59:09 UTC Mon Nov 19 2018
Data     :
0400BA39 E3B35E13 810D8AE5 260B8047 84E8087B 5137319A C2865629 8455928F
D3D9CE39 00E097FF 6CA369C3 EE63BA57 A4C49C02 B408F682 C2153B7F AAE53EF8
A2926001 EF113896 5F1DA056 2D62F292 B860FDFB 0314CE72 F87AA2C9 D5DD29F4
DA85AE4D 1CA453AC 412E911A 419E9B43 0A13DAD3 7B7E88E4 7D96794B 369D6247
E3DA7B8A 5E
```

### Related Topics

Automatic Generation of SSH Host-Key Pairs, on page 158

### Associated Commands

- ssh server algorithms host-key
- show crypto key mypubkey

# Configuring the SSH Client

Perform this task to configure an SSH client.

### SUMMARY STEPS

1. **configure**
2. **ssh client knownhost** *device* **:** /*filename*
3. Use the **commit** or **end** command.
4. **ssh** [**vrf** *vrf-name* ] {*ipv4-address* | *ipv6-address* | *hostname*} [ **username** *user- id*} [ **cipher** **aes** { **128-CTR** | **192-CTR** | **256-CTR** }] **source-interface** *type instance*]

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **ssh client knownhost** *device* **:** /*filename*<br>**Example:**<br>`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.<br>**Note** The complete path of the filename is required. The colon (:) and slash mark (/) are also required. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 4** | **ssh** [**vrf** *vrf-name* ] {*ipv4-address* \| *ipv6-address* \| *hostname*} [ **username** *user- id*} [ **cipher** **aes** { | Enables an outbound SSH connection. |

| | Command or Action | Purpose |
|---|---|---|
| | **128-CTR** \| **192-CTR** \| **256-CTR** }] **source-interface** *type instance*] | • 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.<br>• 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.<br>• The SSHv1 client supports only the 3DES encryption algorithm option, which is still available by default for those SSH clients only.<br>• If the *hostname* argument is used and the host has both IPv4 and IPv6 addresses, the IPv6 address is used. |

- If you are using SSHv1 and your SSH connection is being rejected, the reason could be that the RSA key pair might have been zeroed out or that you have not successfully generated an RSA key pair for your router. Another reason could be that the SSH server to which the user is connecting to using SSHv1 client does not accept SSHv1 connections. Make sure that you have specified a hostname and domain. Then use the **crypto key generate rsa** command to generate an RSA host-key pair, and then enable the SSH server.
- If you are using SSHv2 and your SSH connection is being rejected, the reason could be that the DSA, RSA or ECDSA host-key pair might have been zeroed out. Make sure you follow similar steps as mentioned above to generate the required host-key pairs, and then enable the SSH server.
- When configuring the ECDSA, RSA or DSA key pair, you might encounter the following error messages:
  - No hostname specified

  You must configure a hostname for the router using the **hostname** command.
  - No domain specified

  You must configure a host domain for the router using the **domain-name** command.
- The number of allowable SSH connections is limited to the maximum number of virtual terminal lines configured for the router. Each SSH connection uses a vty resource.
- From Cisco IOS XR Release 6.3.1 onwards, the **ssh client enable cipher** command is added for backward compatibility with the older Cisco IOS XR versions.

  For FIPS compliance, in Cisco IOS XR Releases later than 6.2.1, support for weaker ciphers like 3DES and AES CBC was removed and only AES-CTR cipher is supported.
- SSH uses either local security or the security protocol that is configured through AAA on your router for user authentication. When configuring AAA, you must ensure that the console is not running under AAA by applying a keyword in the global configuration mode to disable AAA on the console.

**Note** If you are using Putty version 0.63 or higher to connect to the SSH client, set the 'Chokes on PuTTYs SSH2 winadj request' option under SSH > Bugs in your Putty configuration to 'On.' This helps avoid a possible breakdown of the session whenever some long output is sent from IOS XR to the Putty client.

# Enabling NETCONF over SSH

This task enables NETCONF over SSH. SSH is currently the only supported transport method .

If the client supports, Netconf over ssh can utilize the multi-channeling capabilities of IOS XR ssh server. For additional details about Multi-channeling in SSH, see *Implementing Secure Shell* in *System Security Configuration Guide*.

**Prerequisites:**

- k9sec pie must be installed, otherwise the port configuration for the netconf ssh server cannot be completed. (The Netconf subsystem for SSH, as well as, SSH cannot be configured without the k9sec pie.)
- Crypto keys must be generated prior to this configuration.
- The Netconf-YANG feature is packaged in the mgbl pie, which must be installed before enabling the Netconf-YANG agent.

## SUMMARY STEPS

1. **configure**
2. **netconf-yang agent ssh**
3. **ssh server netconf** [**vrf** *vrf-name* [**ipv4 access-list***ipv4 access list name*] [**ipv6 access-list** *ipv6 access list name*] ]
4. **ssh server netconf port** *port-number*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **netconf-yang agent ssh**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # netconf agent ssh` | Enables NETCONF agent over SSH connection. After NETCONF is enabled, the Yang model in the controllcker, can configure the relevant models.<br>**Note** The Yang models can be retrieved from the router via NETCONF <get-schema> operation. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **ssh server netconf** [**vrf** *vrf-name* [**ipv4 access-list***ipv4 access list name*] [**ipv6 access-list** *ipv6 access list name*] ]<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # ssh server netconf vrf netconfvrf ipv4 access-list InternetFilter` | Brings up the netconf subsytem support with 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.<br>Optionally ACLs for IPv4 and IPv6 can be used to restrict access to the netconf subsystem of the ssh server before the port is opened.<br>**Note** The netconf subsystem support with SSH server can be configured for use with multiple VRFs . |
| **Step 4** | **ssh server netconf port** *port-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # ssh server netconf port 830` | Configures a port for the netconf ssh server. This command is optional. If no port is specified, port 830 is uses by default.<br>**Note** 830 is the IANA-assigned TCP port for NETCONF over SSH, but it can be changed using this command. |

### What to do next

The **show netconf-yang statistics** command and **show netconf-yang clients** command can be used to verify the configuration details of the netconf agent.

The **clear netconf-yang agent session** command clears the specified Netconf session (on the Netconf server side).

# Configuration Examples for Implementing Secure Shell

This section provides the following configuration example:

## Configuring Secure Shell: Example

This example shows how to configure SSHv2 by creating a hostname, defining a domain name, enabling the SSH server for local and remote authentication on the router by generating a DSA key pair, bringing up the SSH server, and saving the configuration commands to the running configuration file.

After SSH has been configured, the SFTP feature is available on the router.

```
configure
hostname router1
domain name cisco.com
exit
crypto key generate dsa
configure
ssh server
end
```

# Examples: Netconf over SSH

This section illustrates some examples relevant to Netconf:

**Enabling netconf-yang for ssh transport and netconf subsystem for default vrf with default port (830)**

```
config
netconf-yang agent ssh
ssh server netconf vrf default
  !
!
```

**Enabling netconf-yang for ssh transport and netconf subsystem for vrf *green* and vrf *red* with netconf port (831)**

```
config
netconf-yang agent ssh
!
ssh server netconf vrf green
ssh server netconf vrf red
ssh server netconf port 831
  !
!
```

**Show command outputs**

```
show netconf-yang statistics
Summary statistics        requests|             total time|   min time per request|   max
time per request|   avg time per request|
other                              0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
close-session                      4|       0h  0m  0s   3ms|       0h  0m  0s   0ms|
 0h  0m  0s   1ms|       0h  0m  0s   0ms|
kill-session                       0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
get-schema                         0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
get                                0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s
get-config                         1|       0h  0m  0s   1ms|       0h  0m  0s   1ms|
 0h  0m  0s   1ms|       0h  0m  0s   1ms|
edit-config                        3|       0h  0m  0s   2ms|       0h  0m  0s   0ms|
 0h  0m  0s   1ms|       0h  0m  0s   0ms|
commit                             0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
cancel-commit                      0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
lock                               0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
unlock                             0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
discard-changes                    0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|
validate                           0|       0h  0m  0s   0ms|       0h  0m  0s   0ms|
 0h  0m  0s   0ms|       0h  0m  0s   0ms|

show netconf-yang clients
client session ID|   NC version|    client connect time|        last OP time|        last
OP type|    <lock>|
22969|                1.1|         0d  0h  0m  2s|             11:11:24|
close-session|        No|
```

```
15389|               1.1|        0d  0h  0m  1s|           11:11:25|         get-config|
          No|
```

# Multi-channeling in SSH

The multi-channeling (also called multiplexing) feature on the Cisco IOS XR software server allows you to establish multiple channels over the same TCP connection. Thus, rather than opening a new TCP socket for each SSH connection, all the SSH connections are multiplexed into one TCP connection. For example, with multiplexing support on your XR software server, on a single SSH connection you can simultaneously open a pseudo terminal, remotely execute a command and transfer a file using any file transfer protocol. Multiplexing offers the following benefits:

- You are required to authenticate only once at the time of creating the session. After that, all the SSH clients associated with a particular session use the same TCP socket to communicate to the server.
- Saves time consumed otherwise wasted in creating a new connection each time.

Multiplexing is enabled by default in the Cisco IOS XR software server. If your client supports multiplexing, you must explicitly set up multiplexing on the client for it to be able to send multi-channel requests to the server. You can use OpenSSH, Putty, Perl, WinSCP, Putty, FileZilla, TTSSH, Cygwin or any other SSH-based tool to set up multiplexing on the client. Configure Client for Multiplexing, on page 166 provides an example of how you can configure the client for multiplexing using OpenSSH.

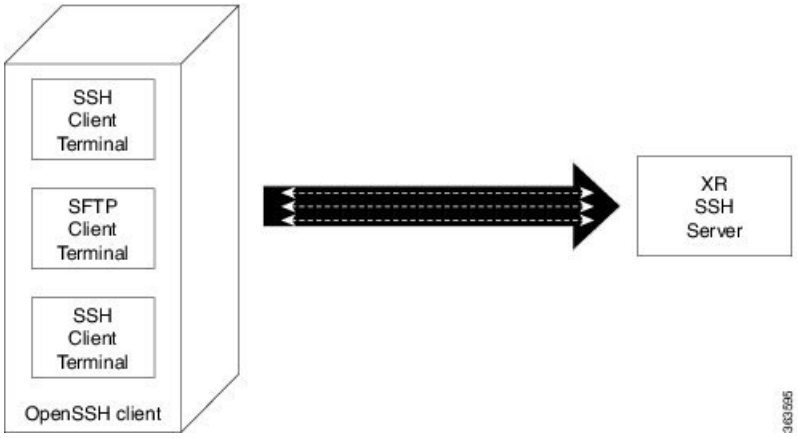
For more information on Multichannel feature, see the Cisco ASR 9000 Series Aggregation Services Router System Security Configuration Guide, Release 5.1.1.

## Restrictions for Multi-channeling Over SSH

- Do not use client multiplexing for heavy transfer of data as the data transfer speed is limited by the TCP speed limit. Hence, for a heavy data transfer it is advised that you run multiple SSH sessions, as the TCP speed limit is per connection.
- Client multiplexing must not be used for more than 15 concurrent channels per session simultaneously.
- You must ensure that the first channel created at the time of establishing the session is always kept alive in order for other channels to remain open.
- The **line template default session-limit** command is not supported for SSH.

## Client and Server Interaction Over Multichannel Connection

The figure below provides an illustration of a client-server interaction over a SSH multichannel connection.




As depicted in the illustration,

- The client multiplexes the collection of channels into a single connection. This allows different operations to be performed on different channels simultaneously. The dotted lines indicate the different channels that are open for a single session.
- After receiving a request from the client to open up a channel, the server processes the request. Each request to open up a channel represents the processing of a single service.

**Note** The Cisco IOX software supports server-side multiplexing only.

# Configure Client for Multiplexing

The SSH client opens up one TCP socket for all the connections. In order to do so, the client multiplexes all the connections into one TCP connection. Authentication happens only once at the time of creating the session. After that, all the SSH clients associated with the particular session uses the same TCP socket to communicate to the server. Use the following steps to configure client multiplexing using OpenSSH:

### SUMMARY STEPS

1. Edit the ssh_config file.
2. Add entries **ControlMaster auto** and **ControlPath**
3. Create a temporary folder.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | Edit the ssh_config file. | Open the ssh_config file with your favorite text editor to configure values for session multiplexing. The system-wide SSH configuration file is located under /etc/ssh/ssh_config. The user configuration file is located under ~/.ssh/config or $HOME/.ssh/config. |
| **Step 2** | Add entries **ControlMaster auto** and **ControlPath**<br>**Example:** | Add the entry ControlMaster auto and ControlPath to the ssh_config file, save it and exit. |

| | Command or Action | Purpose |
|---|---|---|
| | `Host *`<br>`ControlMaster auto`<br>`ControlPath ~/.ssh/tmp/%r@%h:%p` | • ControlMaster determines whether SSH will listen for control connections and what to do about them. Setting the ControlMaster to 'auto' creates a primary session automatically but if there is a primary session already available, subsequent sessions are automatically multiplexed.<br>• ControlPath is the location for the control socket used by the multiplexed sessions. Specifying the ControlPath ensures that any time a connection to a particular server uses the same specified primary connection. |
| **Step 3** | Create a temporary folder. | Create a temporary directory inside the /.ssh folder for storing the control sockets. |

# SSH Configuration Option to Restrict Cipher Public Key and HMAC Algorithm

The Cisco IOS XR software provides a new configuration option to control the key algorithms to be negotiated with the peer while establishing an SSH connection with the router. With this feature, you can enable the insecure SSH algorithms on the SSH server, which are otherwise disabled by default. A new configuration option is also available to restrict the SSH client from choosing the HMAC, or hash-based message authentication codes algorithm while connecting to the SSH server on the router. You can also configure a list of ciphers as the default cipher list, thereby having the flexibility to enable or disable any particular cipher.

Commands introduced:

**Caution** Use caution in enabling the insecure SSH algorithms to avoid any possible security attack.

To disable the HMAC algorithm, use the **ssh client disable hmac** command or **ssh server disable hmac** command in Global Configuration mode.

To enable the required cipher, use the **ssh server enable cipher** command in Global Configuration mode.

The supported encryption algorithms (in the order of preference) are:

1. aes128-ctr
2. aes192-ctr
3. aes256-ctr
4. aes128-gcm@openssh.com
5. aes256-gcm@openssh.com
6. aes128-cbc
7. aes192-cbc

8. aes256-cbc
9. 3des-cbc

In SSH, the CBC-based ciphers are disabled by default. To enable these, you can use the **ssh client enable cipher** command or **ssh server enable cipher** command with the respective CBC options (aes-cbc or 3des-cbc). All CTR-based and GCM-based ciphers are enabled by default.

# Disable HMAC Algorithm

### Configuration Example to Disable HMAC Algorithm

```
Router(config)# ssh server disable hmac hmac-sha1
Router(config)#commit
```

```
Router(config)# ssh client disable hmac hmac-sha1
Router(config)#commit
```

### Running Configuration

```
ssh server disable hmac hmac-sha1
!
```

```
ssh client disable hmac hmac-sha1
!
```

### Related Topics

SSH Configuration Option to Restrict Cipher Public Key and HMAC Algorithm, on page 167

### Associated Commands

- **ssh client disable hmac**
- **ssh server disable hmac**

# Enable Cipher Public Key

### Configuration Example to Enable Cipher Public Key

To enable all ciphers on the client and the server:

Router 1:

```
Router(config)# ssh client algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc
aes128-ctr aes128-cbc aes128-gcm@openssh.com aes256-gcm@openssh.com 3des-cbc
```

Router 2:

```
Router(config)# ssh server algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc
```

```
aes128-ctr aes128-cbc aes128-gcm@openssh.com aes256-gcm@openssh.com 3des-cbc
```

To enable the CTR cipher on the client and the CBC cipher on the server:

Router 1:

```
Router(config)# ssh client algorithms cipher aes128-ctr aes192-ctr aes256-ctr
```

Router 2:

```
Router(config)# ssh server algorithms cipher aes128-cbc aes256-cbc aes192-cbc 3des-cbc
```

Without any cipher on the client and the server:

Router 1:

```
Router(config)# no ssh client  algorithms  cipher
```

Router 2:

```
Router(config)# no ssh server  algorithms  cipher
```

Enable only the deprecated algorithms on the client and the server:

Router 1:

```
Router(config)# ssh client algorithms cipher aes128-cbc aes192-cbc aes256-cbc 3des-cbc
```

Router 2:

```
Router(config)# ssh server algorithms cipher aes128-cbc aes192-cbc aes256-cbc 3des-cbc
```

Enable the deprecated algorithm (using **enable cipher** command) and enable the CTR cipher (using **algorithms cipher** command) on the client and the server:

Router 1:

```
Router(config)# ssh client enable cipher aes-cbc 3des-cbc
Router(config)# ssh client algorithms cipher aes128-ctr aes192-ctr aes256-ctr
```

Router 2:

```
Router(config)# ssh server enable cipher aes-cbc 3des-cbc
Router(config)# ssh server algorithms cipher aes128-ctr aes192-ctr aes256-ctr
```

### Running Configuration

All ciphers enabled on the client and the server:

Router 1:

```
ssh client algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc aes128-ctr aes128-cbc
 aes128-gcm@openssh.com aes256-gcm@openssh.com 3des-cbc
!
```

Router 2:

```
ssh client algorithms cipher aes256-cbc aes256-ctr aes192-ctr aes192-cbc aes128-ctr aes128-cbc
 aes128-gcm@openssh.com aes256-gcm@openssh.com 3des-cbc
!
```

### Related Topics

SSH Configuration Option to Restrict Cipher Public Key and HMAC Algorithm, on page 167

### Associated Commands

- **ssh client enable cipher**
- **ssh server enable cipher**
- **ssh client algorithms cipher**
- **ssh server algorithms cipher**

# Additional References

The following sections provide references related to implementing secure shell.

### Related Documents

| Related Topic | Document Title |
|---|---|
| AAA commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *Authentication, Authorization, and Accounting Commands on* the Cisco IOS XR Software module in *System Security Command Reference for Cisco CRS Routers*. |
| AAA configuration tasks | *Configuring AAA Services on* the Cisco IOS XR Software module in *System Security Configuration Guide for Cisco CRS Routers*. |
| Host services and applications commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *Host Services and Applications Commands on* the Cisco IOS XR Software *Software*module in *IP Addresses and Services Command Reference for Cisco CRS Routers*. |
| IPSec commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *IPSec Network Security Commands on* the Cisco IOS XR Software module in *System Security Command Reference for Cisco CRS Routers* |
| SSH commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *Secure Shell Commands on* the Cisco IOS XR Software module in *System Security Command Reference for Cisco CRS Routers* |

### Standards

| Standards | Title |
|---|---|
| Draft-ietf-secsh-userauth-17.txt | *SSH Authentication Protocol*, July 2003 |
| Draft-ietf-secsh-connect-17.txt | *SSH Connection Protocol*, July 2003 |
| Draft-ietf-secsh-architecture-14.txt | *SSH Protocol Architecture*, July 2003 |
| Draft-ietf-secsh-transport-16.txt | *SSH Transport Layer Protocol*, July 2003 |

### MIBs

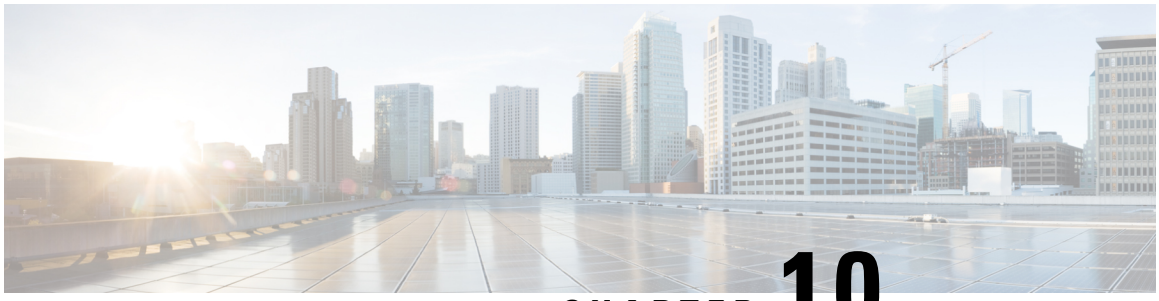
| MIBs | MIBs Link |
|---|---|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml |

### RFCs

| RFCs | Title |
|---|---|
| RFC 6020 | Netconf/ Yang |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 10

# DDoS Mitigation Support on CGSE

Distributed denial-of-service (DDoS) attacks target network infrastructures or computer services resources. The primary goal of DDoS attacks is to deny legitimate users access to a particular computer or network resources, which results in service degradation, loss of reputation, and irretrievable data loss.

DDoS Defense is based on mitigating the attack traffic at entry point into the network.

DDoS Mitigation is the process of detecting increasingly complex and deceptive assaults and mitigating the effects of the attack to ensure business continuity and resource availability.

Threat Management System software is ported to the Cisco Carrier Grade Services Engine (CGSE) to mitigate the attacks and send clean traffic back to the targeted host or network. Cisco CGSE is an integrated multi-CPU service module offering carrier-class performance and scale in support of various applications and services.

For detailed information about DDoS mitigation support concepts, configuration tasks, and examples, see the *Implementing DDoS Mitigation Support on CGSE* module of the *System Security Configuration Guide for Cisco CRS Routers*.



# Implementing DDoS Mitigation Support on CGSE

### What is Distributed Denial-of-service (DDoS)?

Distributed denial-of-service (DDoS) is one in which numerous compromised systems attack a single target system, thereby causing denial of service for users of the targeted system. DDoS attacks target network infrastructures or computer services resources. The primary goal of DDoS attacks is to deny legitimate users access to a particular computer or network resources, which results in service degradation, loss of reputation, and irretrievable data loss.

### What is DDoS Mitigation?

DDoS Mitigation is the process of detecting increasingly-complex and deceptive assaults, and mitigating the effects of such attacks, to ensure business operations continuity and resource availability. DDoS mitigation is based on mitigating the attack traffic at entry point into the network.

Complete DDoS protection provides these benefits:

- Detection and Mitigation of DDoS attacks
- Distinguish good traffic from bad traffic to preserve business continuity
- Include performance and architecture to deploy upstream to protect all points of vulnerability
- Maintain reliable and cost-efficient scalability

### Implementing DDoS Mitigation Support on CGSE

Threat Management System (TMS) software is ported to the Cisco Carrier Grade Services Engine (CGSE) to mitigate the attacks, and to send clean traffic back to the targeted host or network. The Cisco CGSE is a single-slot module supported on all models of Cisco's proven high-end carrier-class routing system: CRS-1 and CRS-3. CGSE offers carrier-class performance and scale in support of various applications and services.

For more information on Implementing DDoS Mitigation Support on CGSE, refer to the *Implementing DDoS Mitigation Support on CGSE* chapter in the *System Security Configuration Guide for Cisco CRS Routers*.

For a complete description of the DDoS Mitigation Support commands, refer to the *DDoS Mitigation Support on CGSE Commands* module of the *System Security Command Reference for Cisco CRS Routers*.

### Feature History for Implementing DDoS Mitigation Support on CGSE

| Release | Modification |
|---|---|
| Release 4.2.3 | This feature was introduced. |

# Restrictions for Implementing DDoS Mitigation

This solution does not provide support for these features on Cisco CRS Router:

- TACACS
- CPU performance evaluation for the TMS–CGSE application
- Latency or Jitter on box performance analysis
- H/W Time stamping of packets
- Co-existence of other services (e.g. CCN, CGN) in the same CGSE blade with TMS–CGSE scrubber
- Incremental routing requirements

# Prerequisites for Implementing DDoS Mitigation

These prerequisites are required to implement DDoS Mitigation support on CGSE:

- Install the CGSE in your Cisco CRS chassis. For information about how to install a CGSE, refer to the *Cisco CRS 20 Gbps Carrier Grade Services Engine Physical Layer Interface Module Installation Note*, made available at URL: http://www.cisco.com/en/US/partner/docs/routers/crs/crs1/plim/installation/guide/20gbpscrscgseplim.html

## Installing and Activating the PIE

The Package Installation Envelope (PIE) files, are installable software files with the .pie extension. PIE files are used to copy one or more software components onto the router. A PIE may contain a single component, a group of components (called a package), or a set of packages (called a composite package).

Use the **show install committed** command in EXEC mode to verify the committed software packages.

You must install and activate the services PIE before you install and use the TMS–CGSE software. Download the *hfr-services-px.pie* to a TFTP server.

For more information about installing PIEs, refer to *Upgrading and Managing Cisco IOS XR Software section* of the *System Management Configuration Guide for Cisco CRS Routers* .

**Note** The TMS–CGSE software is part of a separate image that you download from Cisco.com. For information about the specific images, refer to the *Release Notes for Cisco CRS-1 and Cisco CRS-3 for Cisco IOS XR Software Release 4.2.3.*

### SUMMARY STEPS

1. **admin**
2. **install add** *tftp://<IP address of tftp server>/<location of pie on server>*
3. **install activate** *device:package*
4. **install commit**
5. **exit**
6. **show install committed**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **admin**<br>**Example:**<br>`RP/0/RP0/CPU0:router# admin` | Enters administration EXEC mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **install add** *tftp://<IP address of tftp server>/<location of pie on server>*<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install add tftp://172.201.11.140/auto/tftp-users1/pie/` | Copies the contents of a package installation envelope (PIE) file to a storage device. |
| **Step 3** | **install activate** *device:package*<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install activate disk0:hfr-services-px.pie` | Activates the respective package and adds more functionality to the existing software. |
| **Step 4** | **install commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install commit` | Saves the active software set to be persistent across designated system controller (DSC) reloads. |
| **Step 5** | **exit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# exit` | Exits from the admin mode. |
| **Step 6** | **show install committed**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show install committed` | Shows the list of the committed software packages. |

# Copying TMS-CGSE RPM Package Manager to Route Processor

Perform this task to copy RPM Package Manager (RPM) of TMS–CGSE to Route Processor (RP) disk and to a standby RP.

**Note** RPM Package Manager is a package management system. The name RPM refers to two things: software packaged in the .rpm file format, and the package manager itself. RPM was intended primarily for GNU/Linux distributions; the file format is the baseline package format of the Linux Standard Base.

Copy the TMS-CGSE RPM to a primary RP. You should also copy the RPM to a standby RP to enable TMS to operate in case of a route processor switchover or failover.

We recommend to store the **.rpm** image on a flash card.

### Before you begin

Download the TMS–CGSE RPM image using TFTP, and store it in the "tftp root" directory.

### SUMMARY STEPS

1. **configure**
2. **tftp ipv4 server homedir** *tftp-home-directory*

3. Use the **commit** or **end** command.
4. **copy tftp:**//<*IP address of tftp server*> <*location of TMS–CGSE RPM image on tftp server*>/<*TMS–CGSE RPM image filename* > **disk0:**<*destination filename*>
5. **copy disk0:**<*TMS–CGSE RPM image name*> **location**<*R/S/I of Active RP*> **disk0:location**<*R/S/I of Standby RP*>

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **tftp ipv4 server homedir** *tftp-home-directory*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# tftp ipv4 server homedir disk0` | Enables the TFTP server or a feature running on the TFTP server. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 4** | **copy tftp:**//<*IP address of tftp server*> <*location of TMS–CGSE RPM image on tftp server*>/<*TMS–CGSE RPM image filename* > **disk0:**<*destination filename*><br>**Example:**<br>`RP/0/RP0/CPU0:router# copy tftp://198.51.100.1/tftp_directory/tms-cgse.rpm disk0:tms-cgse.rpm` | Copies the TMS–CGSE RPM image to disk0: |
| **Step 5** | **copy disk0:**<*TMS–CGSE RPM image name*> **location**<*R/S/I of Active RP*> **disk0:location**<*R/S/I of Standby RP*><br>**Example:**<br>`RP/0/RP0/CPU0:router# copy disk0:tms-cgse.rpm location 0/RP0/CPU0 disk0: location 0/RP1/CPU0` | Copies the TMS–CGSE RPM image to the standby RP disk0:<br>**Note** Use **show hfr** command to identify the active RP and standby RP |

# How to Implement DDoS Mitigation Support on CGSE

To implement DDoS Mitigation Support, perform the tasks described in this section. The TMS application hosted on CGSE implements DDoS Mitigation Support on CGSE. Perform these procedures in the order presented to host the TMS application on CGSE.

## Configuring the CGSE Service Role as Service Engine Service Hosting (SESH)

Configure the CGSE service role as Service Engine Service Hosting (SESH). This configuration is done to allow CGSE to start the TMS–CGSE service.

**Important** The removal of the service role is strictly not recommended while the card is active. This puts the card into a FAILED state, and impacts service.

### SUMMARY STEPS

1. **configure**
2. **hw-module service sesh location** <*R/S/I*>
3. Use the **commit** or **end** command.
4. **show running-config service sesh**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **hw-module service sesh location** <*R/S/I*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# hw-module service sesh location 0/1/CPU0` | Configures the service role as SESH for the specified CGSE location in rack/slot/interface format. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **show running-config service sesh**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show running-config service sesh`<br><br>`Wed Jul 11 14:24:31.560 PST`<br>`service sesh sesh1`<br>`service-location preferred-active 0/1/CPU0` | Shows the location of the SESH. |

# Configuring the Service Infrastructure Interface

Configure the service infrastructure (ServiceInfra) interface and associate it with a CGSE module. This configuration needs to be done to send the infrastructure traffic to CGSE and to download the TMS–CGSE. Reboot the CGSE module after ServiceInfra interface configuration and association. Each CGSE should have one ServiceInfra interface.

**Note** The ServiceInfra interface IP address should be configured with a network mask. The maximum limit for this mask is /29. The network should be configured with a minimum of 5 hosts. You must assign a x.x.x.1 IP address. Other ServiceInfra IP addresses do not work for SESH.

### SUMMARY STEPS

1. **configure**
2. **interface ServiceInfra** <*id*>
3. **ipv4 address** <*A.B.C.D*>/<*prefix*>
4. **service-location** <*R/S/I*>
5. Use the **commit** or **end** command.
6. **hw-module location** <*R/S/I*> **reload**
7. **show services role**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br><br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface ServiceInfra** <*id*>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface ServiceInfra 1` | Enters interface configuration mode for the service infrastructure. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **ipv4 address** <*A.B.C.D*>/<*prefix*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# ipv4 address 100.1.1.1/29` | Sets the IP address for this interface. |
| **Step 4** | **service-location** <*R/S/I*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# service-location 0/1/CPU0` | Location of the CGSE you set in **Configuring the Service Role** section in rack/slot/interface format.<br>**Note** To determine where the CGSE modules are installed in the chassis, use the **show platform** command in the EXEC mode. The **show platform** command displays the list of cards that includes CGSE modules with their service location. |
| **Step 5** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 6** | **hw-module location** <*R/S/I*> **reload**<br>**Example:**<br>`RP/0/RP0/CPU0:router# hw-module location 0/1/CPU0 reload` | Reloads CGSE.<br>Use the **show platform** command to monitor the CGSE boot state. The card is fully booted when it switches from the initially BOOTING state to the OK state. |
| **Step 7** | **show services role**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show services role`<br>`Node Configured Role Enacted Role Enabled Services`<br>`----------------------------------------------------`<br>`0/1/CPU0 SESH SESH ServiceInfra` | Displays information about the configured service role. |

# Configuring ServiceEngine–ServiceHost Instance

Configure the SESH instance to run on the CGSE node. The service location specifies the CGSE card location. One active card is supported with no failover, so only the preferred-active argument is supported.

**Note** Before configuring the SESH instance and reloading it, wait approximately 15 minutes for the CGSE to come up in the OK state.

### SUMMARY STEPS

1. **configure**
2. **service sesh** <*name of the sesh instance*>
3. **service-location preferred-active** <*R/S/I*>
4. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service sesh** <*name of the sesh instance*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service sesh sesh1` | Configures service hosting instance. |
| **Step 3** | **service-location preferred-active** <*R/S/I*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-sesh)# service-location preferred-active 0/1/CPU0` | Specifies the CGSE card location in rack/slot/interface format for the SESH instance. Only one active card is supported with no failover. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Configuring Service Application Interfaces

Before configuring the TMS-CGSE software on a CGSE module, configure three Service Application (ServiceApp) interfaces and bind the interfaces with the created SESH instance. Configure one ServiceApp interface for the management path to CGSE. Configure the other two ServiceApp interfaces for the outgoing (offramp) traffic to the TMS-CGSE and for the incoming (onramp) traffic from the TMS-CGSE.

### SUMMARY STEPS

1. **configure**
2. **vrf** *<vrf name>*
3. **commit**
4. **interface ServiceApp** *<ID>*
5. **description** *string*
6. **ipv4 address** *<A.B.C.D>/<prefix>*
7. **service sesh** *<name of the sesh instance>*
8. **interface ServiceApp** *<ID>*
9. **description** *string*
10. **ipv4 address** *<A.B.C.D>/<prefix>*
11. **service sesh** *<name of the sesh instance>*
12. **interface ServiceApp** *<ID>*
13. **description** *string*
14. **vrf** *<vrf name>*
15. **ipv4 address** *<A.B.C.D>/<prefix>*
16. **service sesh** *<name of the sesh instance>*
17. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **vrf** *<vrf name>*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# vrf arbor-tms` | Configures the VRF reference. |
| **Step 3** | **commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# commit` | Use the **commit** command to save the configuration changes to the running configuration file, and remain within the configuration session. |
| **Step 4** | **interface ServiceApp** *<ID>*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface ServiceApp 11` | Enters the interface configuration mode for the service application. |
| **Step 5** | **description** *string*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# description tms1 mgmt interface` | Creates a description for the Service Application Interface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **ipv4 address** <*A.B.C.D*>/<*prefix*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# ipv4 address 10.10.76.1/29` | Sets the IP address for the management interface. |
| **Step 7** | **service sesh** <*name of the sesh instance*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# service sesh sesh1` | Associates the interface with the SESH service instance. |
| **Step 8** | **interface ServiceApp** <*ID*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface ServiceApp 21` | Enters the interface configuration mode for the service application. |
| **Step 9** | **description** *string*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# description tms1 scrb ingress interface` | Creates a description for the Service Application Interface. |
| **Step 10** | **ipv4 address** <*A.B.C.D*>/<*prefix*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# ipv4 address 204.0.0.1/24` | Sets the IP address for the scrubber ingress interface. |
| **Step 11** | **service sesh** <*name of the sesh instance*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# service sesh sesh1` | Associates the interface with the SESH service instance. |
| **Step 12** | **interface ServiceApp** <*ID*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# interface ServiceApp 22` | Enters the interface configuration mode for the service application. |
| **Step 13** | **description** *string*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# description tms1 scrb egress interface` | Creates a description for the ServiceApp interface. |
| **Step 14** | **vrf** <*vrf name*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# vrf arbor-tms` | Places the service interface in VRF.<br>**Note** One ServiceApp interface (either onramp or offramp) must be in VRF to avoid loops. |
| **Step 15** | **ipv4 address** <*A.B.C.D*>/<*prefix*><br>**Example:** | Sets the IP address for the scrubber egress interface. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-if)# ipv4 address 205.0.0.1/24` | |
| **Step 16** | **service sesh** <*name of the sesh instance*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# service sesh sesh1` | Associates the interface with the SESH service instance. |
| **Step 17** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Configuring TMS–CGSE Service and Applications

To enable the TMS–CGSE Service and Applications, configure them first.

Create a Service Engine Service Hosting (SESH) instance and bind the ServiceApp interfaces to CGSE module while configuring TMS-CGSE.

### SUMMARY STEPS

1. **configure**
2. **service sesh** <*name of the sesh instance*>
3. **service-location preferred-active** <*R/S/I*>
4. **service-type** <*service type name*> <*service instance name*>
5. **description** *string*
6. **package** <*name of the TMS–CGSE RPM image* >
7. **application tms-mgmt**
8. **interface ServiceApp** <*ID*>
9. **remote ipv4 address** <*A.B.C.D*>/<*prefix*>
10. exit
11. exit
12. **application tms-scrb**
13. **map ingress-interface ServiceApp**<*ID*> **egress-interface ServiceApp** <*ID*>
14. Use the **commit** or **end** command.
15. **show run service sesh**
16. **show service sesh instance**<name of instance>

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **service sesh** <*name of the sesh instance*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# service sesh sesh1` | Configures service hosting instance. |
| **Step 3** | **service-location preferred-active** <*R/S/I*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-sesh)# service-location preferred-active 0/1/CPU0` | Specifies the CGSE card location in rack/slot/interface format for the SESH instance. Only one active card is supported with no failover. |
| **Step 4** | **service-type** <*service type name*> <*service instance name*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-sesh)# service-type ddos-tms tms1` | Sets the service type. |
| **Step 5** | **description** *string*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-tms1)# description ddos TMS instance 1` | Creates a description for the service. |
| **Step 6** | **package** <*name of the TMS–CGSE RPM image* ><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-tms1)# package tms-cgse.rpm` | Adds the TMS–CGSE image that is part of the instance.<br>**Note** The TMS–CGSE RPM image should be in the tftp_root directory.<br>It takes the TMS–CGSE application approximately 10 minutes to start executing, after committing the configuration. |
| **Step 7** | **application tms-mgmt**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-tms1)# application tms-mgmt` | Specifies the TMS management application. |
| **Step 8** | **interface ServiceApp** <*ID*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-tms-mgmt)# interface ServiceApp 11` | Enters the interface mode of the service application. |
| **Step 9** | **remote ipv4 address** <*A.B.C.D*>/<*prefix*><br>**Example:** | Specifies the remote IPv4 address of the service application. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config-if)# remote ipv4 address 10.10.76.2/29` | **Note** Remote management IP requires a minimum /29 mask. |
| **Step 10** | exit<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-if)# exit` | Exits the Interface configuration mode. |
| **Step 11** | exit<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-tms-mgmt)# exit` | Exits the TMS Management configuration mode. |
| **Step 12** | **application tms-scrb**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-tms1)# application tms-scrb` | Specifies the TMS scrubber application. |
| **Step 13** | **map ingress-interface ServiceApp** <*ID*> **egress-interface ServiceApp** <*ID*><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-tms-scrb)# map ingress-interface ServiceApp 21 egress-interface ServiceApp 22` | Maps the incoming interface and outgoing interface. |
| **Step 14** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 15** | **show run service sesh**<br>**Example:**<br>`RP/0/RP0/CPU0:router# sh running-config service sesh`<br>`service sesh sesh1`<br>`service-location preferred-active 0/1/CPU0`<br>`service-type ddos-tms tms1`<br>` description 'ddos TMS instance 1'`<br>` package arbor-cgse.rpm`<br>` application tms-mgmt`<br>`  interface ServiceApp11`<br>`   remote ipv4 address 10.10.76.2/29`<br>`  !`<br>` !` | Shows the configured parameters. |

| | Command or Action | Purpose |
|---|---|---|
| | application tms-scrb<br>map ingress-interface ServiceApp21<br>egress-interface ServiceApp22 | |
| **Step 16** | **show service sesh instance**<name of instance><br>**Example:**<br>RP/0/RP0/CPU0:router# show services sesh instance all<br><br>Service Infra instance sesh1<br>Application tms1 hosted on Location 0/1/CPU0<br>Octeon 0<br>State - UP - Application Spawned and Service App Interfaces Ready<br>Error Messages - None | Displays the state of the application. Values are:<br>• INIT—Application configuration download is initiated.<br>• WAITING—Application download is complete, but the service application interface is not ready.<br>• UP—Application download is complete, and the service application interface is ready.<br>An error message is displayed when the service application is missing or not configured. |

# Configuring the Zone Secret

Zone Secret is the phrase used by all appliances in the system for internal communication. Zone Secret phrase is required to configure Peakflow SP Leader as Manager of CGSE.

Access TMS–CGSE to configure the Zone Secret.

Refer to the Accessing TMS–CGSE, on page 189 section for the steps for accessing TMS–CGSE.

## SUMMARY STEPS

1. **services tms stop**
2. **services tms secret set** *<zone secret phrase>*
3. **services tms start**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **services tms stop**<br>**Example:**<br>admin@arbos:/# services tms stop | Stops the TMS service in CGSE. |
| **Step 2** | **services tms secret set** *<zone secret phrase>*<br>**Example:**<br>admin@arbos:/# services tms secret set arbor | Sets the Zone Secret phrase. |
| **Step 3** | **services tms start**<br>**Example:**<br>admin@arbos:/# services tms start | Starts the TMS service in CGSE. |

# Configuring Peakflow SP Leader as Manager of CGSE

Peakflow SP leader controls the TMS for all the mitigations. Mitigation is defined in Peakflow SP by the user and Peakflow SP installs the mitigation in TMS.

Configure TMS-CGSE to make Peakflow SP leader the manager of CGSE.

- To enable communication between TMS_CGSE with the Peakflow SP leader
- To enable Peakflow SP leader to control TMS

Access the services line card to configure TMS-CGSE to make Peakflow SP leader the manager of CGSE.

Refer to the Accessing TMS–CGSE, on page 189 section for the steps for accessing TMS–CGSE.

### SUMMARY STEPS

1. **services tms bootstrap** *<Peakflow SP leader IP Address> < Zone secret password>*
2. **config write**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **services tms bootstrap** *<Peakflow SP leader IP Address> < Zone secret password>*<br>**Example:**<br>`admin@arbos:/# services tms bootstrap 121.10.23.1 arbor` | Sets the Peakflow SP Leader the manager of the CGSE module. |
| **Step 2** | **config write**<br>**Example:**<br>`admin@arbos:/# config write` | Saves the configuration changes. |

# Configuring TMS-CGSE in the Peakflow SP Web UI

Peakflow SP Web UI provides interface to configure and manage TMS-CGSE.

Configure TMS-CGSE in the Peakflow SP Web UI.

- To configure TMS-CGSE, select **Administration** > **Peakflow Appliances** from the Peakflow SP Web UI.
- To create a cluster that contains one or more TMS-CGSEs, select **Administration** > **Mitigation** > **TMS-CGSE Clusters** from the Peakflow SP Web UI.

For more information about configuring a TMS-CGSE, see *About Configuring Peakflow SP Appliances* module in the *Peakflow SP User Guide Version 5.7*.

For information about creating a cluster of TMS-CGSEs, see *Configuring TMS-CGSE Clusters* module in the *Peakflow SP User Guide Version 5.7*.

For information about TMS-CGSE deployment scenarios, see *TMS-CGSE Deployment Scenarios* module in the *Peakflow SP User Guide Version 5.7*.

# Accessing TMS–CGSE

To access TMS–CGSE, connect to TMS–CGSE from a Linux server through SSH.

**Note** SSH requires the k9crypto.pie to be installed.

**Note** TMS–CGSE must be reachable from the Linux server. This can be achieved by configuring appropriate routes between the Linux server and the CRS.

### SUMMARY STEPS

1. **ssh tms@** *<TMS Management IP Address>*
2. *Username*
3. *Password*

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **ssh tms@** *<TMS Management IP Address>*<br>**Example:**<br>`eng-1032:~ lnx_server$ ssh tms@10.10.76.2` | Connects to the TMS–CGSE. |
| **Step 2** | *Username*<br>**Example:**<br>`arbos login: admin` | Enter the login username. |
| **Step 3** | *Password*<br>**Example:**<br>`Password:`<br>`admin@arbos:/#` | Enter the Password.<br>**Note** The password will not be visible when entered. |

#### What to do next

It is recommended to change the default password after logging into TMS–CGSE for the first time.

Refer to the Changing TMS–CGSE Login Password, on page 189 section for the steps for changing the username and password.

## Changing TMS–CGSE Login Password

It is recommended to change the default password after logging into TMS–CGSE for the first time.

### SUMMARY STEPS

1. **services tms stop**

2. **services aaa local password** *<old password string>* **interactive**
3. *new password*
4. *new password*
5. **services tms start**

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **services tms stop**<br>**Example:**<br>`admin@arbos:/# services tms stop` | Stops the TMS service in CGSE. |
| **Step 2** | **services aaa local password** *<old password string>* **interactive**<br>**Example:**<br>`admin@arbos:/# services aaa local password admin interactive` | Prompts for password change for the user "admin". |
| **Step 3** | *new password*<br>**Example:**<br>`Password:` | Enter the new password.<br>**Note** The password will not be visible when entered. |
| **Step 4** | *new password*<br>**Example:**<br>`Password:`<br>`admin@arbos:/#` | Re-enter the new password.<br>**Note** The password will not be visible when re-entered. |
| **Step 5** | **services tms start**<br>**Example:**<br>`admin@arbos:/# services tms start` | Starts the TMS service in CGSE. |

## Configuring TMS-CGSE Time Zone and Clock

To configure TMS-CGSE time zone and clock follow these steps.

**SUMMARY STEPS**

1. **system timezone set**
2. *<name of the timezone>*
3. *<name of the sub-timezone>*
4. **clock set** *[MMDDhhmm]*
5. **clock**

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **system timezone set**<br>**Example:**<br>`admin@arbos:/# system timezone set` | Enters the time zone configuration. |
| **Step 2** | *<name of the timezone>*<br>**Example:**<br>`admin@arbos:/# system timezone set`<br>`What timezone are you in? [`?' for list]` **`Asia`** | Sets the time zone. |
| **Step 3** | *<name of the sub-timezone>*<br>**Example:**<br>`admin@arbos:/# system timezone set`<br>`What timezone are you in? [`?' for list] Asia`<br>`Select a sub-timezone [`?' for list]:` **`Calcutta`** | Sets the sub time zone. |
| **Step 4** | **clock set** *[MMDDhhmm]*<br>**Example:**<br>`admin@arbos:/# clock set 01131401`<br>`Fri Jan 13 14:01:00 EST 2012` | Sets the clock. |
| **Step 5** | **clock**<br>**Example:**<br>`admin@arbos:/# clock`<br>`Fri Jan 13 14:22:10 IST 2012` | Displays the clock. |

# Configuration Examples for Implementing DDoS Mitigation Support on CGSE

This section contains the configuration examples for Implementing DDoS Mitigation Support on CGSE.

## Configuring the CGSE Service Role as Service Engine Service Hosting: Example

This example shows how to configure CGSE Service Role as Service Engine Service Hosting (SESH):

```
configure
 hw-module service sesh location 0/1/CPU0
 end

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

```
show running-config service sesh

Wed Jul 11 14:24:31.560 PST
service sesh sesh1
service-location preferred-active 0/1/CPU0
```

# Configuring the Service Infrastructure Interface: Example

This example shows the Service Infrastructure Interface configuration:

```
configure
 interface ServiceInfra 1
 ipv4 address 100.1.1.1/29
 service-location 0/1/CPU0
 end

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

hw-module location 0/1/CPU0 reload

show services role

Node  Configured Role Enacted Role Enabled Services
-------------------------------------------------
0/1/CPU0 SESH           SESH        ServiceInfra
```

# Configuring ServiceEngine–ServiceHost Instance: Example

The following example shows how to configure ServiceEngine–ServiceHost Instance:

```
configure
 service sesh sesh1
  service-location preferred-active 0/1/CPU0
  end
Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:yes
```

# Configuring Service Application Interfaces: Example

This example shows how to configure service application interfaces:

```
configure
 vrf arbor-tms
 commit
 interface ServiceApp 11
  description tms1 mgmt interface
  ipv4 address 10.10.76.1/29
  service sesh sesh1
 interface ServiceApp 21
  description tms1 scrb ingress interface
  ipv4 address 204.0.0.1/24
  service sesh sesh1
 interface ServiceApp 22
```

```
  description tms1 scrb egress interface
  vrf arbor-tms
  ipv4 address 205.0.0.1/24
  service sesh sesh1
  end

Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:yes
```

# Configuring TMS–CGSE Service and Applications: Example

This example shows how to configure TMS–CGSE Service and Applications:

```
configure
 service sesh sesh1
  service-location preferred-active 0/1/CPU0
  service-type ddos-tms tms1
   description ddos TMS instance 1
   package arbor.rpm
   application tms-mgmt
   interface ServiceApp 11
   remote ipv4 address 10.10.76.2/29
   application tms-scrb
   map ingress-interface ServiceApp 21 egress-interface ServiceApp 22
   end
Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:yes
```

This example shows the output of **show run service sesh** command:

```
show running-config service sesh

service sesh sesh1
service-location preferred-active 0/1/CPU0
service-type ddos-tms tms1
 description 'ddos TMS instance 1'
 package arbor-cgse.rpm
 application tms-mgmt
  interface ServiceApp11
   remote ipv4 address 10.10.76.2/29
  !
 !
 application tms-scrb
  map ingress-interface ServiceApp21 egress-interface ServiceApp22
 !
```

This example shows the output of **show service sesh instance** command:

```
show services sesh instance all

Service Infra instance sesh1
Application tms1 hosted on Location 0/1/CPU0
       Octeon 0
       State - UP - Application Spawned and Service App Interfaces Ready
       Error Messages - None
```

# Configuring ACL to Limit Access to CGSE

Access Control Lists(ACLs) are used to restrict access to CGSE modules. You can configure ACLs so that the access to these modules are limited to components of the PeakFlow SP deployment and valid SSH users. To apply this restriction, you have to perform the following tasks:

1. Configure an IPv4 ACL for the Management ServiceApp interface
2. Apply the configured IPv4 ACL to the Management ServiceApp interface.
3. Configure an IPv6 ACL for the Management ServiceApp interface
4. Apply the configured IPv6 ACL to the Management ServiceApp interface.
5. Configure an IPv4 ACL for the ServiceInfra interface.
6. Apply the configured IPv4 ACL to the ServiceInfra interface.

## Configuring an IPv4 ACL for the Management ServiceApp Interface

To configure an IPv4 ACL for the management ServiceApp interface of the CGSE module, follow these steps:

### SUMMARY STEPS

1. **configure**
2. **ipv4 access-list** *access_list_name*
3. **10 permit tcp any host** *tms-cgse_mgmt_IP_address* **eq ssh**
4. **20 permit tcp any eq ident host** *tms-cgse_mgmt_IP_address*
5. **30 permit tcp host** *SP_leader_IP_address* **eq 443 host** *tms-cgse_mgmt_IP address*
6. **40 permit tcp host** *SP_leader_IP_address* *tms-cgse_mgmt_IP address* **eq 443**
7. **50 permit icmp any host** *tms-cgse_mgmt_IP address* **echo**
8. **60 permit icmp any host** *tms-cgse_mgmt_IP address* **echo-reply**
9. **35 permit tcp host** *SP_leader_IP_address* **eq 443 host** *tms-cgse_mgmt_IP address*
10. **45 permit tcp host** *SP_leader_IP_address* *tms-cgse_mgmt_IP address* **eq 443**
11. **30 deny IPv4 any any**
12. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **ipv4 access-list** *access_list_name*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # ipv4 access-list acl` | Creates an IPv4 ACL. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **10 permit tcp any host** *tms-cgse_mgmt_IP_address* **eq ssh**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 10 permit tcp any host 10.10.76.2 eq ssh` | Adds an entry into the IPv4 ACL. This entry permits the SSH access to the TMS management interface CGSE modules . The *tms-cgse_mgmt_IP_address* parameter specifies the IPv4 address chosen for the TMS management interface. |
| **Step 4** | **20 permit tcp any eq ident host** *tms-cgse_mgmt_IP_address*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 20 permit tcp any eq ident host 10.10.76.2` | Adds an entry into the IPv4 ACL. This entry permits the TCP traffic from the ident port( Port number 113) on any host to the TMS management interface on the CGSE modules. The *tms-cgse_mgmt_IP_address* parameter specifies the IPv4 address chosen for the TMS management interface. |
| **Step 5** | **30 permit tcp host** *SP_leader_IP_address* **eq 443 host** *tms-cgse_mgmt_IP address*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 30 permit tcp host 121.10.23.1 eq 443 host 10.10.76.2` | Adds an entry into the IPv4 ACL. This entry permits the TCP traffic from the port 443 on the PeakFlow SP Leader host to the TMS management interface on the CGSE modules. The *SP_leader_IP_address* parameter specifies the IP address assigned to the PeakFlow SP Leader appliance. The *tms-cgse_mgmt_IP_address* parameter specifies the IPv4 address chosen for the TMS management interface. |
| **Step 6** | **40 permit tcp host** *SP_leader_IP_address* *tms-cgse_mgmt_IP address* **eq 443**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 40 permit tcp host 121.10.23.1 10.10.76.2 eq 443` | Adds an entry into the IPv4 ACL. This entry permits the TCP traffic from the PeakFlow SP Leader host to the TMS management interface on port 443 on the CGSE modules. The *SP_leader_IP_address* parameter specifies the IP address assigned to the PeakFlow SP Leader appliance. The *tms-cgse_mgmt_IP_address* parameter specifies the IPv4 address chosen for the TMS management interface.. |
| **Step 7** | **50 permit icmp any host** *tms-cgse_mgmt_IP address* **echo**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 50 permit icmp any host 10.10.76.2 echo` | Adds an entry into the IPv4 ACL. This entry permits the ICMP ping requests to the TMS management interface on the CGSE modules. The *tms-cgse_mgmt_IP_address* parameter specifies the IPv4 address chosen for the TMS management interface. |
| **Step 8** | **60 permit icmp any host** *tms-cgse_mgmt_IP address* **echo-reply**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 60 permit icmp any host 10.10.76.2 echo-reply` | Adds an entry into the IPv4 ACL. This entry permits the ICMP echo replies to the TMS management interface on the CGSE module. The *tms-cgse_mgmt_IP_address* parameter specifies the IPv4 address chosen for the TMS management interface. |
| **Step 9** | **35 permit tcp host** *SP_leader_IP_address* **eq 443 host** *tms-cgse_mgmt_IP address*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 35 permit tcp host 121.10.23.1 eq 443 host 10.10.76.2` | Adds an entry into the IPv4 ACL. This entry permits the traffic from port 443 on the Peakflow SP Leader appliance to the CGSE modules if the Peakflow SP Leader is not the manager appliance. The *SP_leader_IP_address* parameter specifies the IP address assigned to the PeakFlow SP Leader appliance. The *tms-cgse_mgmt_IP_address* |

| | Command or Action | Purpose |
|---|---|---|
| | | parameter specifies the IPv4 address chosen for the TMS management interface. |
| **Step 10** | **45 permit tcp host** *SP_leader_IP_address tms-cgse_mgmt_IP address* **eq 443**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 45 permit tcp host 121.10.23.1 10.10.76.17/29 eq 443` | Adds an entry into the IPv4 ACL. This entry permits the traffic the Peakflow SP Leader appliance to port 443 on the CGSE modules if the Peakflow SP Leader is not the manager appliance. The *SP_leader_IP_address* parameter specifies the IP address assigned to the PeakFlow SP Leader appliance. The *tms-cgse_mgmt_IP_address* parameter specifies the IPv4 address chosen for the TMS management interface. |
| **Step 11** | **30 deny IPv4 any any**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 30 deny IPv4 any any` | Adds an entry into the IPv4 ACL. This entry denies all other access to the CGSE modules except the access given in the previous steps. |
| **Step 12** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

## Applying the IPv4 ACL to the Management Service Application Interface

Perform the following steps to apply the IPv4 Access Control List (ACL) to the Management Service Application (ServiceApp) interface.

### SUMMARY STEPS

1. **configure**
2. **interface ServiceApp** *id*
3. **ipv4 access-group** *access_group_name* **egress**
4. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router# configure` | |
| **Step 2** | **interface ServiceApp** *id*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # interface ServiceApp 11` | Enters the interface mode of the ServiceApp interface. The *id* parameter specifies the number that was assigned to the ServiceApp interface. |
| **Step 3** | **ipv4 access-group** *access_group_name* **egress**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # ipv4 access-group acl egress` | Controls access to the ServiceApp interface towards the outbound direction. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

## Configuring an IPv6 ACL for the Management ServiceApp Interface

To configure an IPv6 ACL for the management Service Application (ServiceApp) interface, follow these steps:

### SUMMARY STEPS

1. **configure**
2. **ipv6 access-list** *access_list_name*
3. **10 deny ipv6 any any log**
4. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **ipv6 access-list** *access_list_name*<br>**Example:** | Creates an IPv6 ACL. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router (config) # ipv6 access-list acl1` | |
| **Step 3** | **10 deny ipv6 any any log**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 10 deny ipv6 any any log` | Adds an entry into the IPv6 ACL. This entry denies all access to the management ServiceApp interface of the CGSE module except the access given in the previous steps. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

## Applying the IPv6 ACL to the Management ServiceApp Interface

Perform the following steps to apply the IPv6 Access Control List (ACL) to the Management ServiceApp interface.

### SUMMARY STEPS

1. **configure**
2. **interface ServiceApp** *id*
3. **ipv6 access-group** *access_group_name* **egress**
4. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface ServiceApp** *id*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # interface ServiceApp 11` | Enters the interface mode of the ServiceApp interface. The *id* parameter specifies the number that was assigned to the ServiceApp interface. |
| **Step 3** | **ipv6 access-group** *access_group_name* **egress**<br>**Example:** | Controls access to the ServiceApp interface towards the outbound direction. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router (config) # ipv6 access-group acl1 egress` | |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

## Configuring an IPv4 ACL for the ServiceInfra Interface

To configure an IPv4 ACL for the Service Infrastructure (ServiceInfra) interface, follow these steps:

### SUMMARY STEPS

1. **configure**
2. **ipv4 access-list** *access_list_name*
3. **10 permit tcp host** *serviceinfra_IP_address* **range 6005 6008** *serviceinfra_subnet/mask*
4. **20 permit tcp host** *serviceinfra_IP_address* **range 16000 16003** *serviceinfra_subnet/mask*
5. **30 permit tcp host** *serviceinfra_IP_address* **range 4000 4003** *serviceinfra_subnet/mask*
6. **40 permit udp host** *serviceinfra_IP_address* **eq 5567** *serviceinfra_subnet/mask*
7. **50 permit icmp any** *serviceinfra_subnet/mask* **echo**
8. **60 permit icmp any** *serviceinfra_subnet/mask* **echo-reply**
9. **70 deny IPv4 any any log**
10. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **ipv4 access-list** *access_list_name*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # ipv4 access-list acl` | Creates an IPv4 ACL. |
| **Step 3** | **10 permit tcp host** *serviceinfra_IP_address* **range 6005 6008** *serviceinfra_subnet/mask* | Adds an entry into the IPv4 ACL. This entry permits TCP traffic from the ServiceInfra interface on ports (6005 to |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`RP/0/RP0/CPU0:router (config) # 10 permit tcp host 100.1.1.1 range 6005 6008 100.1.1.0/29` | 6008) to a host configured on the ServiceInfra subnet on the CGSE modules. The *serviceinfra_IP_address* parameter specifies the IP address assigned to the ServiceInfra interface. The *serviceinfra_subnet/mask* specifies the destination subnet for which traffic will be permitted. |
| **Step 4** | **20 permit tcp host** *serviceinfra_IP_address* **range 16000 16003** *serviceinfra_subnet/mask*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 20 permit tcp host 100.1.1.1 range 16000 16003 100.1.1.0/29` | Adds an entry into the IPv4 ACL. This entry permits TCP traffic from the ServiceInfra interface on ports(16000 to 16003) to a host configured on the ServiceInfra subnet on the CGSE modules. The *serviceinfra_IP_address* parameter specifies the IP address assigned to the ServiceInfra interface. The *serviceinfra_subnet/mask* specifies the destination subnet for which traffic will be permitted. |
| **Step 5** | **30 permit tcp host** *serviceinfra_IP_address* **range 4000 4003** *serviceinfra_subnet/mask*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 30 permit tcp host 100.1.1.1 range 4000 4003 100.1.1.0/29` | Adds an entry into the IPv4 ACL. This entry permits TCP traffic from the ServiceInfra interface on ports(4000 to 4003) to a host configured on the ServiceInfra subnet on the CGSE modules. The *serviceinfra_IP_address* parameter specifies the IP address assigned to the ServiceInfra interface. The *serviceinfra_subnet/mask* specifies the destination subnet for which traffic will be permitted. |
| **Step 6** | **40 permit udp host** *serviceinfra_IP_address* **eq 5567** *serviceinfra_subnet/mask*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 40 permit udp host 100.1.1.1 eq 5567 100.1.1.0/29` | Adds an entry into the IPv4 ACL. This entry permits UDP traffic from the ServiceInfra interface on port 5567 to a host configured on the ServiceInfra subnet on the CGSE modules. The *serviceinfra_IP_address* parameter specifies the IP address assigned to the ServiceInfra interface. The *serviceinfra_subnet/mask* specifies the destination subnet for which traffic will be permitted. |
| **Step 7** | **50 permit icmp any** *serviceinfra_subnet/mask* **echo**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 50 permit icmp any 100.1.1.0/29 echo` | Adds an entry into the IPv4 ACL. This entry permits ICMP ping requests from any source host to the destination host on the *serviceinfra_subnet/mask* subnet. |
| **Step 8** | **60 permit icmp any** *serviceinfra_subnet/mask* **echo-reply**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 60 permit icmp any 100.1.1.0/29 echo-reply` | Adds an entry into the IPv4 ACL. This entry permits ICMP replies from any source host to the destination host on the *serviceinfra_subnet/mask* subnet. |
| **Step 9** | **70 deny IPv4 any any log**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # 70 deny IPv4 any any log` | Adds an entry into the IPv4 ACL. This entry denies all access to the ServiceInfra interface except the access given in the previous steps. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 10** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

## Applying the IPv4 ACL to the Service Infrastructure Interface

Perform the following steps to apply the IPv4 Access Control List (ACL) to the Service Infrastructure (ServiceInfra) interface.

### SUMMARY STEPS

1. **configure**
2. **interface serviceinfra** *id*
3. **ipv4 access-group** *access_group_name* **egress**
4. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **interface serviceinfra** *id*<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # interface serviceinfra 1` | Enters the interface mode of the ServiceInfra interface. The *id* parameter specifies the number that was assigned to the ServiceInfra interface. |
| **Step 3** | **ipv4 access-group** *access_group_name* **egress**<br>**Example:**<br>`RP/0/RP0/CPU0:router (config) # ipv4 access-group acl egress` | Controls access to the ServiceInfra interface towards the outbound direction. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions: |

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

# Additional References

The following sections provide references related to implementing DDoS mitigation support on CGSE.

### Related Documents

| Related Topic | Document Title |
|---|---|
| DDoS Mitigation Support commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *DDoS Mitigation Support commands* on *System Security Command Reference for Cisco CRS Routers* |

### Standards

| Standards | Title |
|---|---|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

### MIBs

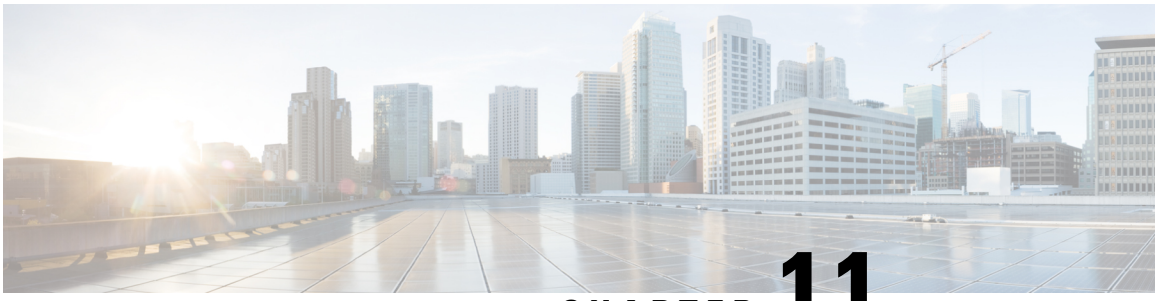
| MIBs | MIBs Link |
|---|---|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml |

### RFCs

| RFCs | Title |
|---|---|
| No new or modified RFCs are supported by this feature. | — |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 11

# Configuring FIPS Mode

The Federal Information Processing Standard (FIPS) 140-2 is an U.S. and Canadian government certification standard that defines requirements that the cryptographic modules must follow. The FIPS specifies best practices for implementing cryptographic algorithms, handling key material and data buffers, and working with the operating system.

In Cisco IOS XR software, these applications are verified for FIPS compliance:

- Secure Shell (SSH)
- Secure Socket Layer (SSL)
- Transport Layer Security (TLS)
- Internet Protocol Security (IPSec) for Open Shortest Path First version 3 (OSPFv3)
- Simple Network Management Protocol version 3 (SNMPv3)
- AAA Password Security

**Note** Any process that uses any of the following cryptographic algorithms is considered non-FIPS compliant:

- Rivest Cipher 4 (RC4)
- Message Digest (MD5)
- Keyed-Hash Message Authentication Code (HMAC) MD5
- Data Encryption Standard (DES)

The Cisco Common Cryptographic Module (C3M) provides cryptographic services to a wide range of the networking and collaboration products of Cisco. This module provides FIPS-validated cryptographic algorithms for services such as RTP, SSH, TLS, 802.1x, and so on. The C3M provides cryptographic primitives and functions for the users to develop any protocol.

By integrating with C3M, the Cisco IOS-XR software is compliant with the FIPS 140-2 standards and can operate in FIPS mode, level 1 compliance.

AAA Password Security for FIPS compliance is available from Cisco IOS XR Software Release Release 6.2.1 and later. See AAA Password Security for FIPS Compliance, on page 14.

# Prerequisites for Configuring FIPS

Install and activate the **hfr-k9sec-px.pie** file.

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command.

If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

### Guidelines for Enabling FIPS Mode

From Cisco IOS XR Software Release 6.7.2 and later, you must follow these guidelines while enabling FIPS mode:

- You must configure the session with a FIPS-approved cryptographic algorithm. A session configured with non-approved cryptographic algorithm for FIPS (such as, **MD5** and **HMAC-MD5**) does not work. This is applicable for OSPF, BGP, RSVP, ISIS, or any application using key chain with non-approved cryptographic algorithm, and only for FIPS mode (that is, when **crypto fips-mode** is configured).
- If you are using any **HMAC-SHA** algorithm for a session, then you must ensure that the configured *key-string* has a minimum length of 14 characters. Otherwise, the session goes down. This is applicable only for FIPS mode.
- If you try to execute the telnet configuration on a system where the FIPS mode is already enabled, then the system rejects the telnet configuration.
- If telnet configuration already exists on the system, and if FIPS mode is enabled later, then the system rejects the telnet connection. But, it does not affect the telnet configuration as such.
- It is recommended to configure the **crypto fips-mode** command first, followed by the commands related to FIPS in a separate commit. The list of commands related to FIPS with non-approved cryptographic algorithms are:
  - **key chain** *key-chain-name* **key** *key-id* **cryptographic-algorithm MD5**
  - **key chain** *key-chain-name* **key** *key-id* **cryptographic-algorithm HMAC-MD5**
  - **router ospfv3 1 authentication ipsec spi 256 md5** *md5-value*
  - **router ospfv3 1 encryption ipsec spi 256 esp des** *des-value*
  - **router ospfv3 1 encryption ipsec spi 256 esp des** *des-value* **authentication md5** *md5-value*
  - **snmp-server user** *username usergroup-name* **v3 auth md5 priv des56**
  - **ssh server algorithms key-exchange diffie-hellman-group1-sha1**
  - **telnet vrf default ipv4 server max-servers** *server-limit*

# Installing and Activating the PIE

The Package Installation Envelope (PIE) files, are installable software files with the .pie extension. PIE files are used to copy one or more software components onto the router. A PIE may contain a single component, a group of components (called a package), or a set of packages (called a composite package).

Use the **show install committed** command in EXEC mode to verify the committed software packages.

You must install and activate the **hfr-k9sec-px.pie** file to configure FIPS. To install and activate the PIE, download the **hfr-k9sec-px.pie** to a TFTP server.

For more information about installing PIEs, refer to *Upgrading and Managing Cisco IOS XR Software section* of the *System Management Configuration Guide for Cisco CRS Routers* .

## SUMMARY STEPS

1. **admin**
2. **install add** *tftp://<IP address of tftp server>/<location of pie on server>*
3. **install activate** *device:package*
4. **install commit**
5. **exit**
6. **show install committed**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **admin**<br>**Example:**<br>`RP/0/RP0/CPU0:router# admin` | Enters administration EXEC mode. |
| **Step 2** | **install add** *tftp://<IP address of tftp server>/<location of pie on server>*<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install add tftp://172.201.11.140/auto/tftp-users1/pie/` | Copies the contents of a package installation envelope (PIE) file to a storage device. |
| **Step 3** | **install activate** *device:package*<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install activate disk0:hfr-k9sec-px.pie` | Activates the respective package and adds more functionality to the existing software. |
| **Step 4** | **install commit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# install commit` | Saves the active software set to be persistent across designated system controller (DSC) reloads. |
| **Step 5** | **exit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)# exit` | Exits from the admin mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **show install committed**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show install committed` | Shows the list of the committed software packages. |

# How to Configure FIPS

Perform these tasks to configure FIPS.

## Enabling FIPS mode

### Before you begin

Refer to the Installing and Activating the PIE, on page 207 section for information on installing and activating the image on the router.

### SUMMARY STEPS

1. **configure**
2. **crypto fips-mode**
3. Use the **commit** or **end** command.
4. **show logging**
5. **admin**
6. **reload location all**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **crypto fips-mode**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)#crypto fips-mode` | Enters FIPS configuration mode.<br>**Note** Stop new incoming SSH sessions while configuring or unconfiguring **crypto fips-mode**. Restart the router upon configuration. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| | | • **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 4** | **show logging**<br>**Example:**<br>`RP/0/RP0/CPU0:router#show logging` | Displays the contents of logging buffers.<br>**Note** Use the **show logging \| i fips** command to filter FIPS specific logging messages. |
| **Step 5** | **admin**<br>**Example:**<br>`RP/0/RP0/CPU0:router#admin` | Enters into the admin EXEC mode. |
| **Step 6** | **reload location all**<br>**Example:**<br>`RP/0/RP0/CPU0:router(admin)#reload location all` | Reloads a node or all nodes on a single chassis or multishelf system. |

# Configuring FIPS-compliant Keys

Perform these steps to configure the FIPS-compliant keys:

### Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 208 section for enabling the FIPS mode.

### SUMMARY STEPS

1. **crypto key generate rsa [usage-keys | general-keys]** *key label*
2. **crypto key generate dsa**
3. **show crypto key mypubkey rsa**
4. **show crypto key mypubkey dsa**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **crypto key generate rsa [usage-keys \| general-keys]** *key label*<br>**Example:**<br>`RP/0/RP0/CPU0:router#crypto key generate rsa general-keys rsakeypair` | Generate a RSA key pair. Ensure that all the key pairs meet the FIPS requirements. The length of the key can vary from 1024 to 2048 bits.<br>The option **usage-keys** generates separate RSA key pairs for signing and encryption. The option **general-keys** generates a general-purpose RSA key pair for signing and encryption.<br>To delete the RSA key pair, use the **crypto key zeroize rsa** *keypair-label* command. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **crypto key generate dsa**<br>**Example:**<br>`RP/0/RP0/CPU0:router#crypto key generate dsa` | Generate a DSA key pair if required. Ensure that all the key pairs meet the FIPS requirements. The length of the key is restricted to1024 bits.<br>To delete the DSA key pair, use the **crypto key zeroize dsa** *keypair-label* command. |
| **Step 3** | **show crypto key mypubkey rsa**<br>**Example:**<br>`RP/0/RP0/CPU0:router#show crypto key mypubkey rsa` | Displays the existing RSA key pairs |
| **Step 4** | **show crypto key mypubkey dsa**<br>**Example:**<br>`RP/0/RP0/CPU0:router#show crypto key mypubkey dsa` | Displays the existing DSA key pairs |

# Configuring FIPS-compliant Key Chain

Perform these steps to configure the FIPS-compliant key chain:

### Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 208 section for enabling the FIPS mode.

### SUMMARY STEPS

1. **configure**
2. **key chain** *key-chain-name*
3. **key** *key-id*
4. **cryptographic-algorithm** {**HMAC-SHA1-20** | **SHA-1**}
5. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router#configure` | Enters the global configuration mode. |
| **Step 2** | **key chain** *key-chain-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)#key chain mykeychain` | Creates a key chain. |
| **Step 3** | **key** *key-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-mykeychain)#key 1` | Creates a key in the key chain. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **cryptographic-algorithm** {**HMAC-SHA1-20** \| **SHA-1**}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-mykeychain-1)#cryptographic-algorithm HMAC-SHA1-20` | Configures the cryptographic algorithm for the key chain. Ensure that the key chain configuration always uses SHA-1 as the hash or keyed hash message authentication code (hmac) algorithm. |
| **Step 5** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Configuring FIPS-compliant Certificates

Perform these steps to configure the FIPS-compliant certificates:

### Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 208 section for enabling the FIPS mode.

## SUMMARY STEPS

1. **configure**
2. **crypto ca trustpoint** *ca-name key label*
3. Use the **commit** or **end** command.
4. **show crypto ca certificates**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **crypto ca trustpoint** *ca-name key label*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)#crypto ca trustpoint msiox rsakeypair` | Configures the trustpoint by utilizing the desired RSA keys.<br>Ensure that the certificates meet the FIPS requirements for key length and signature hash or encryption type. |

| | Command or Action | Purpose |
|---|---|---|
| | | **Note** The minimum key length for RSA or DSA key is 1024 bits. The required hash algorithm is SHA-1-20. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 4** | **show crypto ca certificates**<br>**Example:**<br>`RP/0/RP0/CPU0:router#show crypto ca certificates` | Displays the information about the certificate |

# Configuring FIPS-compliant OSPFv3

Perform these steps to configure the FIPS-compliant OSPFv3:

### Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 208 section for enabling the FIPS mode.

### SUMMARY STEPS

1. **configure**
2. **router ospfv3** *process name*
3. **area** *id*
4. **authentication** {**disable** | **ipsec spi** *spi-value* **sha1** [**clear** | **password**] *password*}
5. **exit**
6. **encryption** {**disable** | {**ipsec spi** *spi-value* **esp** {**3des** | **aes** [**192** | **256**] [**clear** | **password**] *encrypt-password*} [**authentication sha1** [**clear** | **password**] *auth-password*] } }
7. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **router ospfv3** *process name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)#router ospfv3 ospfname` | Configures the OSPFv3 process. |
| **Step 3** | **area** *id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospfv3)#area 1` | Configures the OSPFv3 area ID. The ID can either be a decimal value or an IP address. |
| **Step 4** | **authentication** {**disable** \| **ipsec spi** *spi-value* **sha1** [**clear** \| **password**] *password*}<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospfv3-ar)#authentication ipsec spi 256 sha1 password pa1` | Enables authentication for OSPFv3. Note that the OSPFv3 configuration supports only SHA-1 for authentication. |
| **Step 5** | **exit**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospfv3-ar)#exit` | Exits OSPFv3 area configuration and enters the OSPFv3 configuration mode. |
| **Step 6** | **encryption** {**disable** \| {**ipsec spi** *spi-value* **esp** {**3des** \| **aes** [**192** \| **256**] [**clear** \|**password**] *encrypt-password*} [**authentication sha1** [**clear** \| **password**] *auth-password*] } }<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-ospfv3)#encryption ipsec spi 256 esp 3des password pwd` | Encrypts and authenticates the OSPFv3 packets. Ensure that the OSPFv3 configuration uses the following for encryption in the configuration.<br>• 3DES: Specifies the triple DES algorithm.<br>• AES: Specifies the Advanced Encryption Standard (AES) algorithm.<br>Ensure that SHA1 is chosen if the authentication option is specified. |
| **Step 7** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Configuring FIPS-compliant SNMPv3 Server

Perform these steps to configure the FIPS-compliant SNMPv3 server:

### Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 208 section for enabling the FIPS mode.

### SUMMARY STEPS

1. **configure**
2. **snmp-server user** *username groupname* {**v3** [ **auth sha** {**clear** | **encrypted**} *auth-password* [**priv** {**3des** |**aes** { **128** | **192** | **256**} } {**clear** | **encrypted** } *priv-password*] ] } [**SDROwner** | **SystemOwner**] *access-list-name*
3. Use the **commit** or **end** command.

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router#configure` | Enters the global configuration mode. |
| **Step 2** | **snmp-server user** *username groupname* {**v3** [ **auth sha** {**clear** \| **encrypted**} *auth-password* [**priv** {**3des** \|**aes** { **128** \| **192** \| **256**} } {**clear** \| **encrypted** } *priv-password*] ] } [**SDROwner** \| **SystemOwner**] *access-list-name*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)#snmp-server user user1 g v3 auth sha clear pass priv aes 128 clear privp` | Configures the SNMPv3 server. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Configuring FIPS-compliant SSH Client and Server

Perform these steps to configure the FIPS-compliant SSH Client and the Server:

### Before you begin

Refer the configuration steps in the Enabling FIPS mode, on page 208 section for enabling the FIPS mode.

**SUMMARY STEPS**

1. **ssh** {*ipv4-address* | *ipv6-address*} **cipher aes** {**128-CTR** | **192-CTR** | **256-CTR**} **username** *username*
2. **configure**
3. **ssh server v2**
4. Use the **commit** or **end** command.

**DETAILED STEPS**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **ssh** {*ipv4-address* \| *ipv6-address*} **cipher aes** {**128-CTR** \| **192-CTR** \| **256-CTR**} **username** *username*<br>**Example:**<br>`RP/0/RP0/CPU0:router#ssh 10.1.2.3 cipher aes 128-CTR username user1` | Configures the SSH client. Ensure that SSH client is configured only with the FIPS-approved ciphers. AES(Advanced Encryption Standard)-CTR (Counter mode) is the FIPS-compliant cipher algorithm with key lengths of 128, 192 and 256 bits. |
| **Step 2** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 3** | **ssh server v2**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)#ssh server v2` | Configures the SSH server. |
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Configuration Examples for Configuring FIPS

This section provides examples for configuring FIPS.

## Configuring FIPS: Example

This example shows how to configure FIPS:

```
RP/0/3/CPU0:SSH#configure
RP/0/3/CPU0:SSH(config)#crypto fips-mode
RP/0/3/CPU0:SSH(config)#commit
RP/0/3/CPU0:SSH(config)#end
```

This example shows the output of **show logging** command:

```
RP/0/3/CPU0:SSH(config)#crypto fips-mode
RP/0/3/CPU0:SSH(config)#commit
RP/0/3/CPU0:SSH(config)#end
RP/0/3/CPU0:SSH#show logging

Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
    Console logging: level debugging, 60 messages logged
    Monitor logging: level debugging, 0 messages logged
    Trap logging: level informational, 0 messages logged
    Buffer logging: level debugging, 3 messages logged

Log Buffer (9000000 bytes):
<output omitted>

Log Buffer (307200 bytes):

RP/0/RSP0/CPU0:Apr 16 12:48:17.736 : cepki[433]: The configuration setting for FIPS mode
has been modified. The system must be reloaded to finalize this configuration change. Please
 refer to the IOS XR System Security Configuration Guide, Federal Information Process
Standard(FIPS) Overview section when modifying the FIPS mode setting.
RP/0/RSP0/CPU0:Apr 16 12:48:17.951 : config[65757]: %MGBL-CONFIG-6-DB_COMMIT :
Configuration committed by user 'lab'. Use 'show configuration commit changes 1000000002'
to view the changes.
RP/0/RSP0/CPU0:Apr 16 12:48:23.988 : config[65757]: %MGBL-SYS-5-CONFIG_I : Configured from
 console by lab


....
....
....
```