Configure AAA

This chapter describes the implementation of the administrative model of *task-based authorization* used to control user access in the software system. The major tasks required to implement task-based authorization involve configuring TACACS+ and RADIUS servers and groups.

Note From Release 24.4.1, the AAA local database supports configuring up to 3000 usernames. Although you can configure more than 3000 users, it may impact the system's scale and performance, which are not assured beyond this limit.

Deprecation of Type 7 password and Type 5 secret

Password configuration options before Release 24.4.1

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

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

Deprecation notice

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

- password, on page 2
- masked-password, on page 2
- password-policy, on page 3

- aaa password-policy, on page 4
- secret, on page 4
- masked-secret, on page 5

password

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

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

Changes:

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

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

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

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

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

  show running configuration command output post-upgrade:

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

masked-password

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

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

Changes:

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

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

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

   Converting it to a Type 10 secret for user <user name>.
  ```

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

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

  show running configuration command output post-upgrade:

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

password-policy

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

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

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

Changes:

- All the options that were present until 24.4.1are removed except LINE (to accept cleartext).

- **During upgrade**: Any configuration using the Type 7 password configuration is automatically converted to Type 10 secret.

  Post-upgrade: You can still use the password-policy configurations option after new commits, but the it will be stored as Type 10 secret.

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

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

   Converting it to a Type 10 secret for user <username>.
  ```

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

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

  show running configuration command output post-upgrade:

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

aaa password-policy

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

```
RP/0/RP0/CPU0:ios(config)#aaa password-policy abcd
RP/0/RP0/CPU0:ios(config-pp)#?
min-char-change Number of characters change required between old and new passwords
(deprecated, will be removed in 25.3.1)
restrict-password-advanced Advanced restrictions on new password (deprecated, will be removed
 in 25.3.1)
restrict-password-reverse Restricts the password to be same as reversed old password
(deprecated, will be removed in 25.3.1)
```

Changes:

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

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

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

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

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

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

  show running configuration command output post-upgrade:

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

secret

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

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

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

Changes:

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

  Post-upgrade: Though the keyword 5 has been deprecated, you can still apply the existing configurations using Type 5 secret.
- New **syslog** has been added to indicate the deprecation process:

  ```
  %SECURITY-LOCALD-2-DEPRECATED_SECRET_TYPE : Type 5 secret is deprecated.
  Please use the 'secret' keyword with option type 10 for user.
  ```
- **show running configuration** command output before upgrade:

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

  show running configuration command output post-upgrade:

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

masked-secret

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

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

Changes:

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

  Post-upgrade: Though the keyword 5 has been deprecated, you can still apply the existing configurations using Type 5 masked secret.
- New **syslog** has been added to indicate the deprecation process:

  ```
  %SECURITY-LOCALD-2-DEPRECATED_SECRET_TYPE : Type 5 secret is deprecated.
  Please use the 'secret' keyword with option type 10 for user.
  ```
- **show running configuration** command output before upgrade:

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

show running configuration command output post-upgrade:

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

Special use cases

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

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

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

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

where 'x' is a number representing the index.

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

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

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

About TACACS+

The Terminal Access Controller Access Control System Plus (TACACS+) application is designed to enhance the security of the NCS 1020 device by centralizing user validation. It uses AAA commands and can be enabled and configured on NCS 1020 for improved security. TACACS+ provides detailed accounting information and flexible administrative control over user access.

When TACACS+ server is configured and protocol is enabled on the node, the user credentials are authenticated through TACACS+ server. When the user attempts to log into the node, the username and password is forwarded to the configured TACACS+ servers and get authentication status. If the authentication fails through TACACS+ server, the credentials are sent to the node and are authenticated against the node. If the authentication fails against the node, the user is not allowed to log into the node.

Configure TACACS+ Server

Enabling the AAA accounting feature on a switch allows it to track the network services that users are accessing and the amount of network resources they are using. The switch then sends this user activity data to the TACACS+ security server in the form of accounting records. Each record contains attribute-value pairs and is saved on the security server for analysis. This data can be used for network management, client billing, or auditing purposes.

To configure TACACS+ server, perform these steps:

Procedure

Step 1 Enter into the IOS XR configuration mode.

Example:

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

Step 2 Enable the TACACS+ accounting to send a start-record accounting notice at the beginning of a privileged EXEC process and a stop-record at the end.

Example:

```
RP/0/RP0/CPU0:ios(config)#aaa accounting exec default start-stop group TACACS_ALL
```

Step 3 Create a default command accounting method list for accounting services provided by a TACACS+ security server. This list is configured for privilege level commands and set with a stop-only restriction.

Example:

```
RP/0/RP0/CPU0:ios(config)#aaa accounting exec default start-stop group TACACS_ALL
```

Configure TACACS+ Server Groups

Configuring NCS 1020 to use AAA server groups provides a way to group existing server hosts. This allows you to select a subset of the configured server hosts and use them for a particular service. A server group is used in conjunction with a global server-host list. The server group lists the IP addresses of the selected server hosts.

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).

To configure TACACS+ server groups, perform these steps:

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 configuration, server-private parameters are required.

Procedure

Step 1 Enter into the IOS XR configuration mode.

Example:

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

Step 2 Create an AAA server-group and enter into the server group sub-configuration mode.

Example:

```
RP/0/RP0/CPU0:ios(config)# aaa group server tacacs+ tacgroup1
```

Step 3 Configure the IP address of the private TACACS+ server for the group server.

Example:

```
RP/0/RP0/CPU0:ios(config-sg-tacacs+)# server-private 10.1.1.1 port 49 key a_secret
```

Note

- You can configure a maximum of 10 TACACS+ private servers in a server group.
- If private server parameters are not specified, global configurations are used. If global configurations are not specified, default values are used.

Step 4 Configure the authentication and encryption key used between NCS 1020 and the TACACS+ daemon running on the TACACS+ server. If no key string is specified, the global value is used.

Example:

```
 RP/0/RP0/CPU0:ios(config-sg-tacacs+)# key 7 08984B1A4D0C19157A5F57
```

Step 5 Configure the timeout value that sets the length of time the authentication, authorization, and accounting (AAA) server waits to receive a response from the TACACS+ server.

Example:

```
RP/0/RP0/CPU0:ios(config-sg-tacacs-private)# timeout 4
```

Step 6 Repeat steps 3 to 5 for every private server to be added to the server group.

Step 7 Configure certificate-based authentication for users configured in the TACACS+ server or server groups.

Example:

```
RP/0/RP0/CPU0:ios(config-sg-tacacs-private)#aaa authorization exec default group TACACS_ALL local
```

Step 8 Set the default method list for authentication, and also enables authentication for console in global configuration mode.

Example:

```
RP/0/RP0/CPU0:ios(config-sg-tacacs-private)#aaa authentication login default group TACACS_ALL local
```

Step 9 Commit the changes and exit all the configuration modes.

commit

end

Step 10 Verify the TACACS+ server group configuration details.

Example:

```
RP/0/RP0/CPU0:ios# show tacacs server-groups
```

About RADIUS

Remote Authentication Dial-In User Service (RADIUS) is a distributed client/server system that provides security against unauthorized access in distributed client/server networks. In Cisco's implementation, RADIUS clients operate on Cisco NCS 1020 and send requests for authentication and accounting to a central RADIUS server that contains all user authentication and network service access information.

Cisco's AAA security paradigm supports RADIUS, which can be used alongside other security protocols like TACACS+, Kerberos, and local username lookup.

Configure RADIUS Server Groups

You 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).

You can configure a maximum of 30 servers and private servers each per RADIUS server group. To configure RADIUS server groups, perform these tasks:

Before you begin

Ensure that the external server is accessible at the time of configuration.

Procedure

Step 1 **configure**

Example:

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

Enters mode.

Step 2 **aaa group server radius** *group-name*

Example:

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

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

Step 3 **radius-server** {*ip-address*}

Example:

```
RP/0/RP0/CPU0:ios(config)# radius-server host 192.168.20.0
```

Specifies the hostname or IP address of the RADIUS server host.

Step 4 **auth-port** *port-number*

Example:

```
RP/0/RP0/CPU0:ios(config)#auth-port 1812
```

Specifies the User Datagram Protocol (UDP) destination port for authentication requests; the host is not used for authentication if set to 0. If unspecified, the port number defaults to 1645.

Step 5 **acct-port** *port-number*

Example:

```
RP/0/RP0/CPU0:ios(config)# acct-port 1813
```

Specifies the UDP destination port for accounting requests; the host is not used for accounting if set to 0. If unspecified, the port number defaults to 1646.

Step 6 **key** *string*

Example:

```
RP/0/RP0/CPU0:ios(config-radius-host)#key 7 08984B1A4D0C19157A5F57
```

Specifies the authentication and encryption key used between NCS 1020 and the RADIUS server. This key overrides the global setting of the **radius-server key** command. If no key string is specified, the global value is used.

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. This is because the leading spaces are ignored, but spaces within and at the end of the key are used. If you use spaces in the key, do not enclose the key in quotation marks unless the quotation marks themselves are part of the key.

Step 7 Repeat steps 4 to 6 for every external radius server to be added to the server group.

—

Step 8 **aaa authentication**{**login**}{**default**} **group** *group-name* **local**

Example:

```
RP/0/RP0/CPU0:ios(config-radius-host)#aaa authentication login default group radius local
```

Specifies the default method list for authentication, and also enables authentication for console in global configuration mode.

Step 9 Use the **commit** or **end** command.

Step 10 **show radius server-groups** [*group-name* [**detail**]]

Example:

```
RP/0/RP0/CPU0:ios# show radius server-groups
```

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