802-1x Port Based Authentication

- 802.1X port-based authentication, on page 1
- Usage guidelines and restrictions for 802.1X port-based authentication, on page 4
- Understanding 802.1X Port-Based Authentication, on page 5
- Configure 802.1X host-modes, on page 6
- Configure 802.1X with remote RADIUS authentication, on page 6
- Configure 802.1X with local EAP authentication, on page 8
- Configure router as 802.1X supplicant, on page 9
- Verify 802.1X Port-Based Authentication, on page 10
- 802.1X port-based authentication with MAB fallback, on page 12
- RADIUS Change of Authorization for 802.1X and MAB sessions, on page 16

802.1X port-based authentication

The 802.1X port-based authentication is a type of authentication that enables port-based network access control as defined by the IEEE 802.1X standard that

- operates at the Layer 2 and prevents an unauthorized network device from connecting to a network via LAN ports,
- uses a client-server model to authenticate a client network device (hereafter referred to as client) before allowing it access to a network, and
- controls port access, ensuring the port-interface blocks all traffic to and from a client until it is successfully authenticated.

In 802.1X port-based authentication, the client-server model involves a client device requesting network access by communicating with an authentication server, typically a Remote Authentication Dial-In User Service (RADIUS) server. The client sends its credentials, which the server verifies against authorized user databases. Upon successful authentication, the server permits network access, ensuring only authorized devices can connect.

This chapter describes how to configure IEEE 802.1X port-based authentication in Cisco 8000 series routers to prevent unauthorized network devices from gaining access to the network.

Table 1: Feature History Table

Feature Name	Release Information	Feature Description
802.1X port-based authentication	Release 26.1.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q200, P100], 8700 [ASIC: P100, K100], Modular Systems (8800 [LC ASIC: Q100, Q200, P100])(select variants only*) *This feature is extended to these hardware variants: • 8201 • 8201-32FH • 88-LC0-36FH-M • 8212-48FH-M • 8711-32FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM • 8712-MOD-M • 8711-48Z-M • 8011-4G24Y4H-I • 8011-32Y8L2H2FH • 8011-12G12X4Y-A

Feature Name	Release Information	Feature Description
802.1X port-based authentication	Release 25.1.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100], 8700 [ASIC: P100]); Modular Systems (8800 [LC ASIC: P100]) You can now secure network access by requiring client network devices to authenticate with encrypted digital certificates before gaining access. The 802.1X port-based authentication ensures that a port remains closed to all traffic until the connected client successfully completes authentication using the Extensible Authentication Protocol with TLS (EAP-TLS) encryption. This prevents unauthorized access and enforces secure, certificate-based communication, enhancing network security and integrity.

Supported protocols for 802.1X port-based authentication

802.1X port-based authentication supports these protocols for secure and reliable network access:

- **EAP** —Extensible Authentication Protocol (EAP) is a flexible authentication framework that supports multiple authentication methods, allowing network devices to negotiate the most appropriate protocol for secure communications.
- **EAP-TLS**—Extensible Authentication Protocol with Transport Layer Security (EAP-TLS) is an authentication protocol that leverages TLS to provide robust encryption and mutual authentication, ensuring secure and private communication between the client and server during the authentication process.
- **EAPOL**—Extensible Authentication Protocol over LAN (EAPOL) is a network protocol used in the IEEE 802.1X standard, facilitating the exchange of EAP packets over wired or wireless LANs, enabling secure port-based network access control.

IEEE 802.1X Device Roles

Devices in the network have specific roles in IEEE 802.1X authentication:

- **Authenticator**—A router that facilitates authentication for other network devices or clients on the same LAN.
- **Supplicant**—A network device or client that seeks authentication from an authenticator on a point-to-point LAN segment.

- **Authentication server**—A RADUIS server that verifies client credentials and authorizes network access through the authenticator.

802.1X host modes

The 802.1X host modes table describes the two host modes supported by 802.1X.

For information on how to configure the host modes, refer to Configure 802.1X host-modes.

Table 2: 802.1X host modes

Host modes	Description
Single-host	While in this mode, the port allows a single host or client to be authenticated and allows only ingress traffic from the authenticated peer. A security violation is detected if more than one client is present.
Multi-auth	This is the default host mode. While in this mode, multiple hosts can independently authenticate through the same port and ingress traffic is allowed from all authenticated peers. The router can support up to 20 clients using the 802.1X protocol in multi-authentication mode.

Prerequisites for 802.1X Port-Based Authentication

Prerequisites for 802.1X port-based authentication are:

- K9sec RPM is required to enable this feature.
- Ensure that both RADIUS/EAP-server and supplicant are configured with supported EAP methods when remote authentication is used.
- If the device is used as a local EAP server, only EAP-TLS method is supported.
- Ensure that a Certificate Authority (CA) server is configured for the network with a valid certificate.
- Ensure that the supplicant, authenticator, and CA server are synchronized using Network Time Protocol (NTP). If time is not synchronized on all these devices, certificates may not be validated.

Usage guidelines and restrictions for 802.1X port-based authentication

Consider these restrictions and usage guidelines when implementing 802.1X port-based authentication on the Cisco 8000 platform:

Port authentication

- 802.1X port authentication must be configured on physical ports.
- Supported modes for 802.1X port-based authentication:
    - Single-host

- Multi-auth

VLAN sub-interfaces

- VLAN sub-interfaces must have pre-configured VLAN IDs.
- All VLAN-tagged traffic is dropped until successful 802.1X authentication of the port.
- No default VLAN assignment is provided for unauthenticated MAC addresses.
- Authenticated MAC addresses are validated at the main port, independent of VLAN assignment.
- VLAN-tagged traffic is allowed only for authenticated MAC addresses.

Untagged traffic

- Untagged EAPOL traffic is always allowed.
- All other untagged traffic is dropped until successful 802.1X authentication of the port.
- Untagged traffic is allowed only for authenticated MAC addresses.
- No default VLAN assignment is provided for untagged traffic by the port.

Unsupported hardware variants

802.1X port-based authentication is not supported on these hardware variants.

- 8608-SYS
- 8404-RSP1-48-EM
- 8404-RSP1-2FH/4H

Understanding 802.1X Port-Based Authentication

IEEE 802.1X port-based authentication is configured on Cisco 8000 series router to prevent unauthorized routers (supplicants) from gaining access to the network. An authentication server validates the supplicant that is connected to an authenticator port, before the services offered by the client or the network is made available to the supplicant.

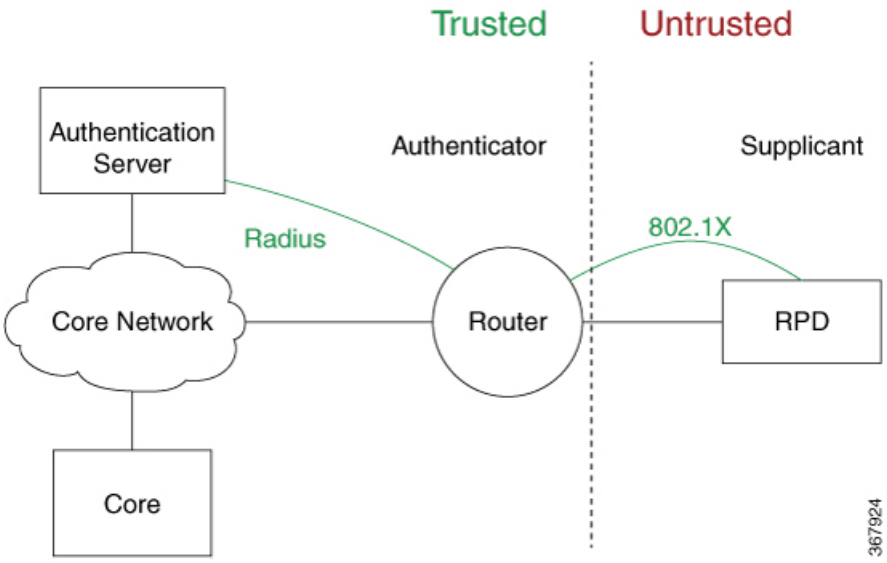
Until the supplicant is authenticated, the port is in *Unauthorized* state, and 802.1X access control allows only Extensible Authentication Protocol over LAN (EAPOL) packets through the port. EAPoL frames can have either default EtherType of 0x888E or Cisco-defined EtherType of 0x876F. After successful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

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

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

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

Figure 1: Topology for IEEE 802.1X Port-Based Authentication



By default, the dot1x configured port is in multi-auth mode. However, this behaviour can be altered by changing the host mode under dot1x profile.

Note Port-control is enforced only on the ingress traffic.

Configure 802.1X host-modes

Use the following steps to configure 802.1X host-modes. Here, `host-mode` is introduced under the authenticator mode in dot1x profile. The default is `multi-auth` mode.

```
Router# configure terminal
Router(config)# dot1x profile {name}
Router(config-dot1x-auth)# pae {authenticator}
Router(config-dot1x-auth-auth)# host-mode
  multi-auth   multiple authentication mode
  multi-host   multiple host mode
  single-host  single host mode
```

Configure 802.1X with remote RADIUS authentication

Use this procedure to configure 802.1X port-based authentication using a remote RADIUS server. This method enables centralized authentication and access control, ensuring network security.

Before you begin

Before you configure 802.1X port-based authentication using a remote RADIUS server, verify

- the RADIUS server is operational and reachable, and
- the pre-shared key for secure communication between the device and the RADIUS server is available.

Procedure

Step 1 Configure the RADIUS server.

Example:

```
Router# configure terminal
         Router(config)# radius-server host 209.165.200.225 auth-port 1646 key secret007
         Router(config)# radius-server vsa attribute ignore unknown
         Router(config)# commit
```

Verify the configuration using the **show run radius** command.

Step 2 Configure the 802.1X authentication method.

Example:

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

Note
Only default AAA method is supported for 802.1X authentication.

Verify the configuration using the **show run aaa** command.

Step 3 Configure the 802.1X authenticator profile.

Example:

```
Router(config)# dot1x profile auth
         Router(config-dot1x-auth)# pae authenticator
         Router(config-dot1x-auth)# authenticator
         Router(config-dot1x-auth-auth)# timer reauth-time 3600
         Router(config-dot1x-auth-auth)# host-mode {multi-auth | single-host}
         Router(config-dot1x-auth-auth)# commit
```

Verify the configuration using the **show run dot1x** command.

Step 4 Attach the 802.1X profile to an interface.

Example:

```
Router(config)# interface HundredGigE 0/3/0/0
         Router(config-if)# dot1x profile auth
         Router(config-if)# commit
```

Verify the configuration using the **show run interface HundredGigE 0/3/0/0** command.

The port now uses 802.1X EAP-TLS authentication to validate connected devices via the remote RADIUS server. Unauthorized devices cannot access the network until successfully authenticated.

Configure 802.1X with local EAP authentication

Use this procedure to configure 802.1X port-based authentication using a locally hosted EAP server on a Cisco 8000 router. This configuration enables mutual authentication between the router and client using certificates.

In local EAP authentication, the EAP-server is co-located with the authenticator locally on the router. This feature enables the router to authenticate 802.1X clients with EAP-TLS method using TLS Version 1.2. It provides EAP-TLS based mutual authentication, where a Master Session Key (MSK) is generated on successful authentication.

Procedure

Step 1 Generate an RSA key pair.

Example:

```
Router# crypto key generate rsa <keypair-label>
```

Step 2 Configure a trustpoint.

Example:

```
Router(config)# crypto ca trustpoint <tp_name>
         Router(config-trustp)# enrollment url <ca-url>
         Router(config-trustp)# subject-name <x.500-name>
         Router(config-trustp)# rsakeypair <keypair-label>
         Router(config-trustp)# commit
```

Trustpoints let you manage and track CAs and certificates. A trustpoint includes the identity of the CA, CA-specific configuration parameters, and an association with one, enrolled identity certificate. After you have defined a trustpoint, you can reference it by name in commands requiring that you specify a CA.

Step 3 Configure a domain name.

Example:

```
Router(config)# domain name <domain-name>
```

The domain name is required for certificate enrollment.

Step 4 Configure certificates.

Example:

```
Router# crypto ca authenticate <tp_name>
         Router# crypto ca enroll <tp_name>
```

Ensure the Certificate Authority (CA) issues the required certificates for authentication.

Step 5 Configure an EAP profile.

Example:

```
Router(config)# eap profile <profile_name>
Router(config-eap)# identity <user-name>
Router(config-eap)# method tls pki-trustpoint <tp_name>
Router(config-eap)# commit
```

Step 6 Configure the 802.1X authenticator profile.

Example:

```
Router(config)# dot1x profile local_auth
Router(config-dot1x-auth)# pae authenticator
Router(config-dot1x-auth)# authenticator
Router(config-dot1x-auth-auth)# eap profile <profile_name>
Router(config-dot1x-auth-auth)# host-mode {multi-auth | single-host}
Router(config-dot1x-auth-auth)# timer reauth-time 3600
Router(config-dot1x-auth-auth)# commit
```

Step 7 Configure 802.1X profile on an interface.

Example:

```
Router(config)# interface <interface-name>
         Router(config-if)# dot1x profile local_auth
         Router(config-if)# commit
```

The router now uses 802.1X with a local EAP server to authenticate connected devices using EAP-TLS and certificates.

Configure router as 802.1X supplicant

Before you begin

Before you configure the router as a 802.1X supplicant

- generate an RSA key pair for certificate-based authentication,
- configure a trustpoint with the appropriate Certificate Authority (CA),
- configure a domain name,
- obtain CA certificate for the given trust point and enroll the device certificate with CA, and
- ensure an EAP profile is configured for authentication.

Procedure

Step 1 Configure the 802.1X supplicant profile.

Example:

```
Router(config)# dot1x profile supp
Router(config-dot1x-supp)# pae supplicant
Router(config-dot1x-supp)# supplicant
Router(config-dot1x-supp-supp)# eap profile <profile_name>
Router(config-dot1x-supp-supp)# commit
```

Step 2 Attach the supplicant profile to an interface.

Example:

```
Router(config)# interface <interface-name>
Router(config-if)# dot1x profile supp
Router(config-if)# commit
```

The router is now configured as a supplicant in 802.1X authentication, enabling it to authenticate with an upstream authenticator using EAP-TLS and certificates.

Verify 802.1X Port-Based Authentication

The 802.1X authentication can be verified using the following:

- Show command outputs
- Syslog messages

Show Command Outputs

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

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

Dot1x info for HundredGigE 0/0/1/0
-----------------------------------------------------------------
Interface short name      : Hu 0/0/1/0
Interface handle          : 0x4080
Interface MAC             : 021a.9eeb.6a59
Ethertype                 : 888E
PAE                       : Authenticator
Dot1x Port Status         : AUTHORIZED
Dot1x Profile             : test_prof
L2 Transport              : FALSE
Authenticator:
   Port Control           : Enabled
   Config Dependency      : Resolved
   Eap profile            : None
   ReAuth                 : Disabled
Client List:
```

```
      Supplicant          : 027e.15f2.cae7
 Programming Status       : Add Success
      Auth SM State       : Authenticated
      Auth Bend SM State  : Idle
      Last authen time    : 2018 Dec 11 17:00:30.912
      Last authen server  : 10.77.132.66
      Time to next reauth : 0 day(s), 00:51:39
MKA Interface:
   Dot1x Tie Break Role   : NA (Only applicable for PAE role both)
   EAP Based Macsec       : Disabled
   MKA Start time         : NA
   MKA Stop time          : NA
   MKA Response time      : NA
```

Syslog Messages

Syslogs on Authenticator

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

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

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

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


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

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

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

When authentication fails, the following syslog messages are displayed:

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

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

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

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

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

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

Syslogs on Supplicant

```
%L2-DOT1X-5-SUPP_SUCCESS : Hu0/0/1/0 : Authentication successful with authenticator
008a.96a4.b050

%L2-DOT1X-5-SUPP_FAIL : Hu0/0/1/0 : Authentication successful with authenticator
0000.0000.0000.0000

%L2-DOT1X-5-SUPP_FAIL : Hu0/0/1/0 : Authentication successful with authenticator
008a.96a4.b028
```

802.1X port-based authentication with MAB fallback

MAC Authentication Bypass (MAB) is a network fallback authentication method that

- applies to networks with both 802.1X-capable and non-802.1X-capable devices
- authenticates devices that cannot use 802.1X,
- allows 802.1X to take precedence over MAB, and
- terminates the MAB authorization upon successful 802.1X authentication.

Table 3: Feature History Table

Feature Name	Release Information	Feature Description
MAC Authentication Bypass fallback method for 802.1X authentication	Release 25.3.1	Introduced in this release on: Fixed Systems (8200 [ASIC: P100]); Modular Systems (8800 [LC ASIC: Q200, P100]) (select variants only*) You can use MAC Authentication Bypass (MAB) as a fallback method to enhance network security and flexibility when routers do not support the 802.1X protocol. By default, 802.1X authentication is set as the primary authentication method. In multi-authentication mode, a router supports up to 20 MAB clients simultaneously, in networks with a mix of 802.1X-capable and non-802.1X-capable devices. The feature introduces these changes: CLI: **show dot1x port authentication** *This feature is supported on: • 8212-48FH-M • 88-LC0-36FH-M • 88-LC1-36EH • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

Methods used to prioritize 802.1X authentication

An 802.1X-enabled port running in fallback MAB mode can prioritize 802.1X authentication using one of these methods:

- EAPOL detection: In MAB mode, the router monitors for EAPOL packets from a supplicant. If EAPOL packets are detected and 802.1X authentication succeeds, the 802.1X session takes precedence, terminating MAB authorization and assuming control.
- Change of Authorization (CoA): The AAA server triggers a CoA request through RADIUS to enforce 802.1X authentication.

Conditions for MAB fallback from 802.1X authentication

Each condition outlines a specific scenario and the corresponding router response that triggers fallback to MAB.

- Initial authentication: If no EAPOL packets are detected within three intervals of 10 seconds each.
- Reauthentication: If no EAPOL packets are detected within three intervals of 30 seconds each.
- Authentication failure: If 802.1X authentication failure is reported by the remote server.
- Authentication timeout: When the RADIUS server times out during 802.1X authentication, and the **server dead action auth-fail** command is explicitly set under the dot1x profile. Although the command is enabled by default, explicitly configuring it ensures that the intended action remains consistent across software updates or device replacements. This reduces the risk of unintended changes in behavior.

Authentication failure scenarios of 802.1X and MAB

Authentication failure scenarios that help to understand the common failure reasons and the router's response to them.

- 802.1X authentication failure scenarios
- MAB authentication failure scenarios

802.1X authentication failure scenarios

- No EAPoL received: The client either does not support 802.1X or has not responded to the 802.1X EAPoL request.
- Server reject: The RADIUS server sent an *Access-Reject* message for 802.1X authentication, indicating authentication failure.
- Server unreachable: When the RADIUS server is unavailable, and the default **server dead action auth-fail** command is applied.

Behavior on 802.1X authentication failure

- No EAPoL received: The client either does not support 802.1X or has not responded to the 802.1X EAPoL request.

- Server reject: The RADIUS server sent an *Access-Reject* message for 802.1X authentication, indicating authentication failure.
- Server unreachable: When the RADIUS server is unavailable, and the default **server dead action auth-fail** command is applied.

MAB authentication failure scenarios

This section outlines common RADIUS server failures affecting MAB authentication.

- Server reject: The RADIUS server sent an *Access-Reject* message for MAB authentication, indicating authentication failure.
- Server unreachable: When the RADIUS server is unavailable, and the default **server dead action auth-fail** command is applied.

Behavior on MAB method failure

If all authentication methods have failed, after 60 seconds, the client and any associated programming will be deleted.

Configure 802.1X port-based authentication with MAB fallback

The purpose of this task is to configure 802.1X and MAB authentication methods and verify the result of each authentication method attempted.

Procedure

Step 1 Configure the port to enforce 802.1x and MAB authentication before allowing access to various services through the port.

Example:

```
Router#configure
Router(config)#dot1x profile auth_mab
Router(config-dot1x-auth_mab)#pae authenticator
Router(config-dot1x-auth_mab)#mab
Router(config-dot1x-auth_mab)#authenticator timer reauth-time 60
Router(config-dot1x-auth_mab)#authenticator server dead action auth-fail
```

The 802.1x profile named **auth_mab** is configured globally with **pae authenticator** and **mab** commands.

Step 2 Execute the **show run dot1x profile** <*proflle-name*> command to verify if the port is configured to authenticate both 802.1X and MAB.

Example:

```
Router#show run dot1x profile auth_mab
Mon Jun  9 14:20:38.956 IST
dot1x profile auth_mab
mab
pae authenticator
authenticator
timer reauth-time 60
```

```
server dead action auth-retry
!
!
```

Step 3 Enable the port control on the interface by configuring the 802.1x profile under it.

Example:

```
Router#configure
Router(config)#interface GigabitEthernet 0/1/0/0
Router(config-if)#dot1x profile auth_mab
```

Step 4 Execute the **show run interface** <*interface*> command to verify that the port control is configured on the interface.

Example:

```
Router#show run interface GigabitEthernet 0/1/0/0
Mon Jun  9 14:23:13.982 IST
interface GigabitEthernet0/1/0/0
 dot1x profile auth_mab
!
```

Step 5 Verify that the output shows 802.1X authentication failed because no EAPoL packets were received from the client.

Example:

```
Router#show dot1x port authentication
Thu Feb  6 05:50:00.086 UTC

NODE: node0_2_CPU0
=============================================================================
     Interface             Client          Method        Status
=============================================================================

GigabitEthernet 0/1/0/0  ac4a.6730.0620     mab        Authorized


Router#show dot1x port authentication detail
Thu Feb 6 05:42:37.691 UTC

NODE: node0_2_CPU0

Port Authentication Info for GigabitEthernet 0/1/0/0
-----------------------------------------------------------------
Interface Handle         : 0x80001c0
Interface State          : Up
Port Status              : Authorized (1/1)
Profile                  : test_auth_mab
Method List              : dot1x, mab
Client :
MAC Address              : ac4a.6730.0620
Status                   : Authorized
Programming Status       : Add Success
Unauthorized Timer       : 60s, timer off
Method:
dot1x                    : Failed (No EAPoL received)
mab                      : Success
```

a) Verify that the output shows the client is authenticated through the 802.1X authentication method.

Example:

```
Router#show dot1x port authentication
```

```
Thu Feb  6 05:50:00.086 UTC

NODE: node0_2_CPU0
================================================================================
     Interface             Client          Method        Status
================================================================================

GigabitEthernet 0/1/0/0  ac4a.6730.0620     dot1x        Authorized
```

```
Router#show dot1x port authentication detail
Thu Feb 6 05:59:41.206 UTC

NODE: node0_2_CPU0

Port Authentication Info for GigabitEthernet 0/1/0/0
----------------------------------------------------------------
Interface Handle         : 0x80001c0
Interface State          : Up
Port Status              : Authorized (1/1)
Profile                  : test_auth_mab
Method List              : dot1x, mab
Client :
MAC Address              : ac4a.6730.0620
Status                   : Authorized
Programming Status       : Add Success
Unauthorized Timer       : 60s, timer off
Method:
dot1x                    : Success
mab                      : Not Run
```

RADIUS Change of Authorization for 802.1X and MAB sessions

RADIUS Change of Authorization (CoA) is an authentication, authorization, and accounting (AAA) feature that

- provides a mechanism to dynamically change the attributes of the active client sessions on the router using CoA requests
- allows CoA clients such as external AAA or policy servers to initiate re-authentication requests to the device that is functioning as the CoA server, such as a Cisco IOS XR router
- supports CoA requests per session for re-authentication of 802.1X and MAC authentication bypass (MAB) sessions using the RADIUS protocol, and
- eliminates the need for complete session termination for changing the session attributes of active client sessions.

CoA using RADIUS protocol is an AAA feature that allows you to change the client session attributes after authentication. When a policy changes for a user or user group in AAA, administrators can send the RADIUS CoA packets from the AAA server to reinitialize authentication and apply the new policy.

Table 4: Feature History Table

Feature Name	Release Information	Feature Description
RADIUS Change of Authorization for 802.1X and MAB sessions	Release 25.3.1	Introduced in this release on: Fixed Systems (8200 [ASIC: Q100, Q200], 8700 [ASIC: K100]); 8010 [ASIC: A100]); Modular Systems (8800 [LC ASIC: Q100, Q200]) You can now prioritize dot1x authentication on a dot1x-enabled port that is already running in fallback MAC authentication bypass (MAB) mode on your router. This feature allows Change of Authorization (CoA) for 802.1X- or MAB-authenticated clients through RADIUS CoA requests from the external AAA server or policy server. This capability allows external servers to trigger authentication updates for the client sessions without the need to disconnect and reconnect. The feature introduces these changes: CLI: • The **show dot1x port authentication** command is modified to include the **statistics** keyword. • The **show radius dynamic-author** command is modified to include the **Dot1x COA Statistics** field in the output.

RADIUS CoA requests for 802.1X and MAB sessions

Cisco IOS XR Software supports the RADIUS CoA extensions defined in RFC 5176 to support CoA for 802.1X and MAB sessions. These CoA extensions are typically used in a push model in which the request originates from the external server to the device attached to the network. It enables dynamic reconfiguration of sessions from external AAA or policy servers to allow session identification, host re-authentication, and session termination.

Response codes for CoA requests

Two response codes are possible with the push model's `CoA-Request`:

- CoA acknowledgment (CoA-ACK)
- CoA non-acknowledgment (CoA-NAK)

Error strings in the CoA-NAK message

These are the possible error strings in the CoA-NAK message:

- Client not found
- Client not authorized
- Invalid Request
- Invalid NAS port ID
- Resource unavailable at the moment, try after sometime
- Missing mandatory subscriber:command=reauthenticate
- Missing mandatory Calling-Station-Id
- Invalid subscriber command received
- Calling-Station-Id format is invalid

These are the error strings in the CoA-NAK message that are specific to rate limit:

- Per-client CoA rate limit: 1 request every 60s
- Global CoA rate limit: 128 requests per minute

CoA commands and rules for 802.1X and MAB session re-authentication

To initiate session re-authentication, the AAA server must send a standard CoA-Request message containing specific Vendor-Specific Attributes (VSAs) listed in this table.

CoA command	Cisco VSA
session reauthenticate	**Cisco:Avpair="subscriber:command=reauthenticate"**
session reauthenticate-type	**Cisco:Avpair="subscriber:reauthenticate-type=last"** or **Cisco:Avpair="subscriber:reauthenticate-type=rerun"**

The **reauthenticate-type** defines the type of authentication method to be used for re-authentication. It can be either of these:

- **last**: The CoA re-authentication request uses the authentication method that last succeeded on the session
- **re-run**: The authentication process is completely rerun

These rules apply to 802.1X and MAB session re-authentication using RADIUS CoA requests:

- The **subscriber:command=reauthenticate** must be present in the VSA to trigger a re-authentication.

- If **subscriber:reauthenticate-type** is not specified, the behavior is same as that of **reauthenticate-type=last**. That is, the system reruns the previous successful authentication method for that session, by default.
- The **subscriber:reauthenticate-type** is valid only when included with the **subscriber:command=reauthenticate** VSA.

Session identification attributes in the CoA requests

Session identification attributes included in the CoA message must match the session with the router. Else, the router returns a CoA-NAK message.

Attribute	IETF attribute	Importance
Calling-Station-Id	Attribute 31, which contains the host MAC address.	Mandatory
NAS-Port-ID	Attribute 87, which contains the complete interface name on which the client MAC address is currently served.	Optional

Guideline for supporting RADIUS CoA for 802.1X and MAB sessions

Explicitly configure the CoA server with the IP address of the CoA client and the associated pre-shared key to ensure secure communication.

The router supports only pre-shared key–based RADIUS CoA.

Enable RADIUS CoA for 802.1X and MAB sessions

The purpose of this task is to enable RADIUS CoA for 802.1X and MAB sessions on Cisco IOS XR routers.

Before you begin

Configure CoA server with the IP address of the CoA client and the associated pre-shared key to ensure secure communication.

Procedure

Step 1 Enable RADIUS CoA server on the router.

Example:

```
Router#configure
Router(config)#aaa server radius dynamic-author
Router(config-dynamic-author)#port 1700
Router(config-dynamic-author)#server-key test-password
Router(config-dynamic-author)#client 10.101.130.122 vrf default
Router(config-dynamic-author-client)#server-key test-password
Router(config-dynamic-author-client)#commit
```

Step 2 Set up the remote RADIUS server on the router.

Example:

```
Router#configure
Router(config)#radius-server vsa attribute ignore unknown
Router(config)#radius-server host 10.101.130.122 auth-port 1812 acct-port 1813
Router(config-radius-host)#key test-password
Router(config-radius-host)#commit
```

Step 3 Send a CoA request with Cisco VSAs from the CoA client to change authorization for an existing 802.1X or MAB session.

The CoA client can be an external AAA server or policy server.

Step 4 Check the system logs on the router to verify if the change of authorization for the client session succeeded.

Table 5: System logs for CoA-ACK scenarios

If the client is...	And re-authentication type is...	Then the router displays..
802.1X-authenticated	**last**	`%L2-DOT1X-5-COA_ACK : Hu0/0/0/5: Processed the CoA to 'reauthenticate(last)' client 008a.96a4.b052` `%L2-DOT1X-5-AUTH_SUCCESS : Hu0/0/0/5: Authentication successful for client 008a.96a4.b052`
802.1X-authenticated	**rerun**	`%L2-DOT1X-5-COA_ACK : Hu0/0/0/5: Processed the CoA to 'reauthenticate(rerun)' client 008a.96a4.b052` `%L2-DOT1X-5-AUTH_SUCCESS : Hu0/0/0/5: Authentication successful for client 008a.96a4.b052`
MAB-authenticated	**last**	`%L2-DOT1X-5-COA_ACK : Hu0/0/0/5: Processed the CoA to 'reauthenticate(last)' client f4db.e62e.c61b` `%L2-DOT1X-5-MAB_AUTH_SUCCESS : Hu0/0/0/5: Authentication successful for client f4db.e62e.c61b`

MAB-authenticated	**rerun**	`%L2-DOT1X-5-COA_ACK : Hu0/0/0/5: Processed the CoA to 'reauthenticate(rerun)' client f4db.e62e.c61b`
		The router processes 802.1X authentication first. If it fails, it proceeds with the re-authentication process using MAB.
		`%L2-DOT1X-5-MAB_AUTH_SUCCESS : Hu0/0/0/5: Authentication successful for client f4db.e62e.c61b`

Table 6: System logs for CoA-NAK scenarios

If the client is...	And re-authentication type is...	Then the router displays..
802.1X- or MAB-unauthorized	**last**	%L2-DOT1X-5-COA_NAK : Hu0/0/0/5:008a.9634.5cd4: Rejected the CoA to 'reauthenticate(last)' because of 'Client not authorized'

If the client is...	And re-authentication type is...	Then the router displays..
802.1X- or MAB-unauthorized	**rerun**	%L2-DOT1X-5-COA_NAK : Hu0/0/0/5:008a.9634.5cd4: Rejected the CoA to 'reauthenticate(rerun)' because of 'Client not authorized'