Controller Self-Signed Certificate for Wireless AP Join

Controller self-signed certificates for wireless AP Join

A controller self-signed certificate is a digital certificate generated and issued by a wireless controller itself that

- enables secure CAPWAP communication for access point (AP) join processes
- is used when manufacturer installed SUDI certificates are unavailable or incompatible, and
- helps ensure connectivity and secure onboarding of APs under specific scenarios.

Self-signed certificates allow Cisco Catalyst controllers to fulfill certificate requirements for AP joining and management, especially when manufacturer-installed or third-party certificates are missing or cannot be validated.

Use cases for controller self-signed certificate for wireless AP join

You need controller self-signed certificates in specific circumstances to enable wireless APs to join the controller successfully.

Controller self-signed certificate use cases

These situations require controller self-signed certificates:

- The Cisco Catalyst 9800-CL platform is not equipped with manufacturer-installed SUDI certificates. Configure self-signed certificates on these controllers.
- Some APs running earlier software and using a Manufacturer Installed Certificate (MIC) issued by a SHA1 Cisco Trusted CA cannot join controllers with SHA2 SUDI certificates. During the CAPWAP join process, these APs display a bad certificate error and fail the DTLS handshake.

Workaround: To support AP upgrades, configure a self-signed certificate for the controller as a workaround for the second use case. After updating all APs, delete the self-signed certificates. Then, revert to the SUDI certificate.

Note This workaround does not apply to Embedded Wireless Controllers running on Catalyst 9000 series switches. It is only valid for hardware appliance controllers such as Cisco Catalyst 9800-40, 9800-80, and 9800-L.

Requirement: Use minimum RSA key size for DTLS connections

Always ensure certificates used for DTLS connections (for AP and mobility) have an RSA key size of at least 2048 bits. Using a smaller key size will cause AP and mobility connections to fail after a reload.

This principle applies to all DTLS connections involving access points (AP) and mobility features that require device certificates.

Using a minimum RSA key size of 2048 bits ensures compliance with security best practices and prevents connection failures due to insufficient key strength.

DTLS connections for AP and mobility will remain operational after a reload when certificates meet the minimum RSA key size requirement.

To verify the device certificate key size, use this command:

```
show crypto pki certificate verbose tp-name
```

Replace *tp-name* with your trustpoint name.

Prerequisites for controller self-signed certificate for wireless AP join

- Ensure that the VLAN interface is up and that the IP address is reachable.
- Ensure that the **ip http server** is enabled. For more information, see Enabling HTTP Server .
- Set the **clock calendar-valid** command appropriately. For more information, see *Configure clock calendar.*
- Check whether the PKI CA server is configured. If it is, delete the existing CA server configuration.

Note The **show crypto pki server** command output should not display anything.

Configure clock calendar (CLI)

Set the clock calendar to ensure accurate timekeeping on the device.

This configuration is necessary for time-sensitive operations and logging on the device.

Before you begin

Access to the device with appropriate privileges to enter global configuration mode.

Follow these steps to configure clock calendar:

Procedure

Step 1 Enter global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Enable clock calendar.

Example:

```
Device(config)# clock calendar-valid
```

Step 3 Exit configuration mode.

Example:

```
Device(config)# exit
```

The clock calendar is now configured and operational on the device.

Enable HTTP server using CLI

Enable the HTTP server to allow web-based management of the device.

This configuration is applicable for devices that require HTTP access for management purposes.

Before you begin

Ensure you have administrative access to the device.

Procedure

Step 1 Enter global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Enable the HTTP server on your IP or IPv6 system.

Example:

```
Device(config)# ip http server
```

Enables the HTTP server on your IP or IPv6 system, including a Cisco web browser user interface. By default, the HTTP server uses the standard port 80.

Step 3 Enable the secure HTTP server on your IP or IPv6 system.

Example:

```
Device(config)# ip http secure-server
```

Enables the secure HTTP server on your IP or IPv6 system, providing encrypted access to the web interface.

Step 4 Exit configuration mode.

Example:

```
Device(config)# exit
```

The HTTP server is now enabled, allowing web-based management of the device.

Configure CA server (CLI)

Deploy a CA server to manage certificates for devices in the network.

This configuration is essential for environments that require secure communication through certificates.

Procedure

Step 1 Enter global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Generate RSA keys for the CA server.

Example:

```
Device(config)# crypto key generate rsa general-keys modulus size_of_key_module label
keypair_name
```

Configures a certificate for the controller.

When you generate RSA keys, you are prompted to enter a modulus length. A longer modulus length might be more secure, but it takes longer to generate and to use.

Note
The recommended key-pair name is *WLC_CA* and key modulus is *2048* bits.

Step 3 Enable the IOS certificate server.

Example:

```
Device(config)# crypto pki server certificate_server_name
```

Enables IOS certificate server.

Note
The *certificate_server_name* must be the same name as the *keypair_name*.

Step 4 Configure the issuer name for the CA certificate.

Example:

```
Device(cs-server)# issuer-name O=Cisco Virtual Wireless LAN Controller, CN=CA-vWLC
```

Configures X.509 distinguished name for the issuer CA certificate.

Note
You need to configure the same *issuer-name* as suggested for AP join.

Step 5 Grant certificate requests automatically.

Example:

```
Device(cs-server)# grant auto
```

Grants certificate requests automatically.

Step 6 (Optional) Specify the hash function for the signature used in the granted certificates.

Example:

```
Device(cs-server)# hash sha256
```

Specifies the hash function for the signature used in the granted certificates.

Step 7 (Optional) Specify the lifetime in days of a CA certificate.

Example:

```
Device(cs-server)# lifetime ca-certificate time-interval
```

Specifies the lifetime in days of a CA certificate.

Step 8 (Optional) Specify the lifetime in days of a granted certificate.

Example:

```
Device(cs-server)# lifetime certificate time-interval
```

Specifies the lifetime in days of a granted certificate.

Step 9 Set the CA key and CA certificate archive format and password.

Example:

```
Device(cs-server)# database archive pkcs12 password 0 cisco123
```

Sets the CA key and CA certificate archive format and password to encrypt the file.

Step 10 Enable the certificate server.

Example:

```
Device(cs-server)# no shutdown
```

Enables the certificate server.

Note
Issue this command only after you have completely configured your certificate server.

Step 11 Return to privileged EXEC mode.

Example:

```
Device(cs-server)# end
```

Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

The CA server is now configured and ready to manage certificates for devices in the network.

Configure trustpoint (CLI)

Configure a trustpoint to enable secure communication with a Certificate Authority (CA).

This configuration is necessary for devices that require secure communications and certificate management.

Before you begin

Ensure you have access to the device and the necessary permissions to configure trustpoints.

Follow these steps to configure trustpoint using CLI commands:

Procedure

Step 1 Enter global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Generate RSA keys for the trustpoint.

Example:

```
Device(config)# crypto key generate rsa exportable general-keys modulus
size-of-the-key-modulus label label
```

When you generate RSA keys, you are prompted to enter a modulus length. A longer modulus length might be more secure, but it takes longer to generate and to use.

Step 3 Create a new trustpoint for an external CA server.

Example:

```
Device(config)# crypto pki trustpoint trustpoint_name
```

Here, *trustpoint_name* refers to the trustpoint name.

Note
Ensure that same names are used for key-pair (*label*) and *trustpoint_name*.

Step 4 Map RSA key with that of the trustpoint.

Example:

```
Device(ca-trustpoint)# rsakeypair RSA_key
```

Maps RSA key with that of the trustpoint.

- RSA_key—Refers to the RSA key pair label.
- key_size—Refers to the signature key length. The value ranges from 360 to 4096.

Step 5 Create subject name parameters for the trustpoint.

Example:

```
Device(ca-trustpoint)# subject-name subject_name
```

Specifies the subject name for the trustpoint.

Step 6 Check revocation.

Example:

```
Device(ca-trustpoint)# revocation-check none
```

Specifies whether to check for certificate revocation.

Step 7 Specify the hash algorithm.

Example:

```
Device(ca-trustpoint)# hash sha256
```

Specifies the hash algorithm to be used.

Step 8 Specify the serial number.

Example:

```
Device(ca-trustpoint)# serial-number
```

Specifies the serial number for the trustpoint.

Step 9 (Optional) Set certificate key-usage purpose.

Example:

```
Device(ca-trustpoint)# eku request server-auth client-auth
```

Sets the extended key usage for the certificate.

Step 10 Enable password for the trustpoint.

Example:

```
Device(ca-trustpoint)# password 0 password
```

Step 11 Enroll the URL for the CA server.

Example:

```
Device(ca-trustpoint)# enrollment url url
```

Replace the dummy IP with management VLAN interface IP of the controller where CA server is configured.

Step 12 Exit the configuration mode.

Example:

```
Device(ca-trustpoint)# end
```

The trustpoint is now configured and ready for use with the CA server.

Authenticate and enroll the PKI trustpoint with CA server

Authenticate and enroll the PKI trustpoint to establish a secure connection with the Certification Authority (CA) server.

This procedure is used in environments where a PKI trustpoint needs to be authenticated and enrolled with a Certification Authority (CA) server for certificate management.

Before you begin

Ensure that the Certification Authority (CA) server is reachable and the trustpoint is properly configured.

Follow these steps to authenticate and enroll the PKI trustpoint with CA server:

Procedure

Step 1 Enter global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Authenticate the PKI trustpoint with the CA server.

Example:

```
Device(config)# crypto pki authenticate trustpoint-name
Certificate has the following attributes:
Fingerprint MD5: 64C5FC9A C581D827 C25FC3CF 1A7F42AC
Fingerprint SHA1: 6FAFF812 7C552783
6A8FB566 52D95849 CC2FC050
% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
```

Fetches the CA certificate.

Step 3 Enroll the PKI trustpoint for a client certificate.

Example:

```
Device(config)# crypto pki enroll trustpoint-name
Enter following answers for UI interaction:
% Include an IP address in the subject name? [no]: no
Request certificate from CA? [yes/no]: yes
```

Step 4 Exit global configuration mode.

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode. Alternatively, you can also press Ctrl-Z to exit global configuration mode.

The PKI trustpoint is now authenticated and enrolled with the Certification Authority (CA) server, allowing for secure certificate management.

Tag wireless management trustpoint name

Tag a wireless management trustpoint name for identification and management purposes.

This configuration is used in environments where wireless management trustpoints need to be clearly identified for management and security.

Before you begin

Ensure you have the necessary privileges to enter global configuration mode.

Procedure

	Command or Action	Purpose
Step 1	Enter global configuration mode. **Example:** `Device# configure terminal`	
Step 2	Tag the wireless management trustpoint name. **Example:** `Device(config)# wireless management trustpoint` *`trustpoint-name`*	
Step 3	Exit global configuration mode. **Example:** `Device(config)# end`	Returns to privileged EXEC mode. Alternatively, you can also press Ctrl-Z to exit global configuration mode.

The wireless management trustpoint name is successfully tagged and can be used for management purposes.

Disable PKI server

Disable the PKI server to prevent it from issuing certificates.

This procedure is used in scenarios where the PKI server is no longer needed or must be temporarily disabled.

Procedure

Step 1 Enter global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Create a new server for an external CA server.

Example:

```
Device(config)# crypto pki server certificate-server-name
```

The *certificate-server-name* refers to the server name.

Note
Here, server_name refers to the server name.

Step 3 Disable the certificate server.

Example:

```
Device(cs-server)# shutdown
```

Update purpose string to \

Step 4 Return to privileged EXEC mode.

Example:

```
Device(cs-server)# end
```

Alternatively, you can also press Ctrl-Z to exit global configuration mode.

The PKI server is now disabled and will not issue certificates.

Verify controller certificates for wireless AP join

Use these commands to verify controller certificates for wireless access point (AP) join.

To view the CA server details, use the command:

```
Device# show crypto pki server
Certificate Server WLC_CA:
Status: enabled
State: enabled
Server's configuration is locked (enter "shut" to unlock it)
Issuer name: O=Cisco Virtual Wireless LAN Controller, CN=CA-vWLC
CA cert fingerprint: 79A3DBD5 59A7E384 73ABD152 C133F4E2
Granting mode is: auto
Last certificate issued serial number (hex): 1
CA certificate expiration timer: 12:04:00 UTC Mar 8 2029
CRL NextUpdate timer: 18:04:00 UTC Mar 11 2019
Current primary storage dir: nvram:
Database Level: Minimum - no cert data written to storage
```

To view the trustpoint details, use this command:

```
Device# show crypto pki trustpoint ewlc-tp1 status
Trustpoint ewlc-tp1:
...
State:
Keys generated ..... Yes (General Purpose, exportable)
Issuing CA authenticated ..... Yes
Certificate request(s) ..... Yes
```

To view the wireless management trustpoint details, use this command:

```
Device# do show wireless management trustpoint
Trustpoint Name : ewlc-tp1
Certificate Info : Available
Certificate Type : SSC
Certificate Hash : 4a5d777c5b2071c17faef376febc08398702184e
Private key Info : Available
FIPS suitability : Not Applicable
```

To view the Hypertext Transfer Protocol (HTTP) server status, use this command:

```
Device# show ip http server status | include server status
HTTP server status: Enabled
HTTP secure server status: Enabled
```