Configure Multi-level CA on OpenSSL to Generate IOS XE Certificates

Contents

Introduction

This document describes a method to create a multi-level CA to create general purpose certificates compatible with Cisco IOS® XE devices.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- How to use the OpenSSL application.
- Public Key Infrastructure (PKI) and digital certificates.

Components Used

The information in this document is based on these software and hardware versions:

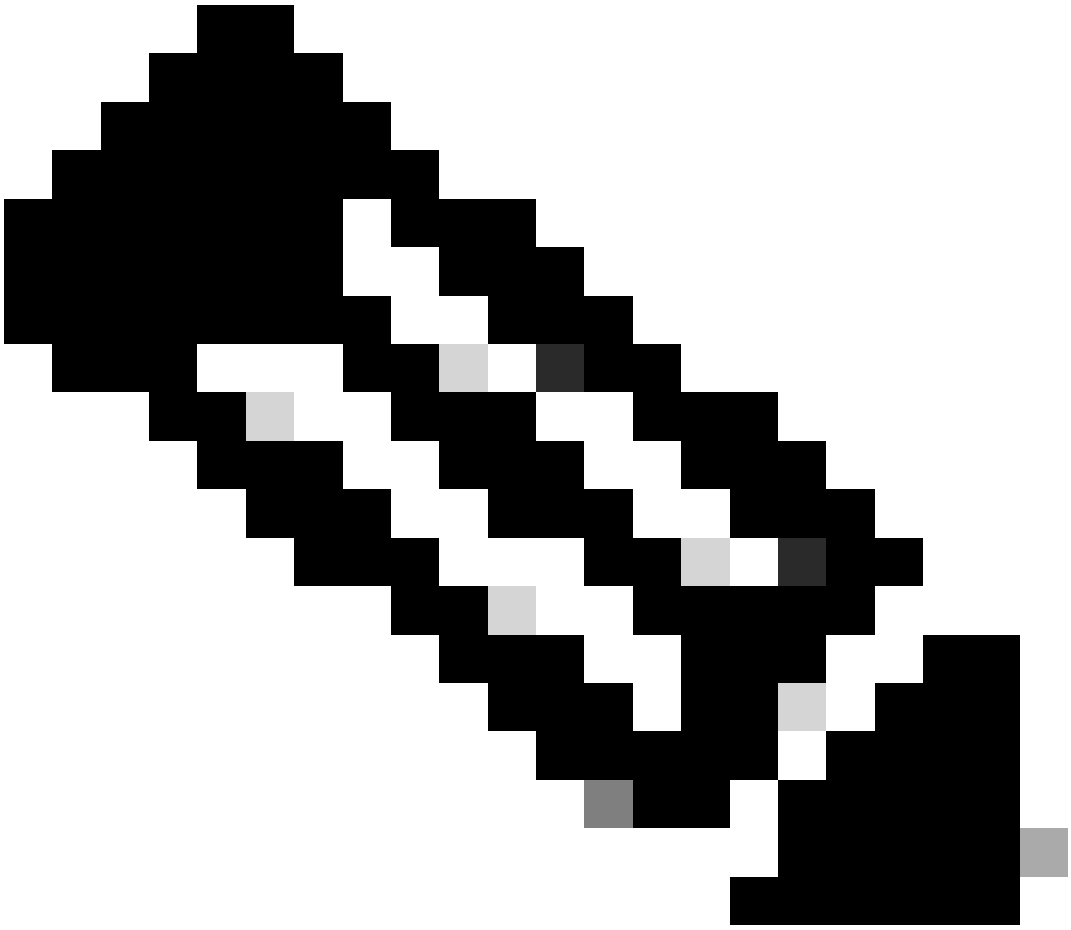
- OpenSSL application (version 3.0.2).
- 9800 WLC (Cisco IOS XE version 17.12.3).

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Configure

Overview

The purpose is to create a two level, local Certificate Authority (CA) with a Root CA and an Intermediate CA to sign device certificates. Once the certificates are signed, they are imported to the Cisco IOS XE device.



> **Note**: This document uses Linux specific commands to create and arrange files. The commands are explained so you can perform the same action on other operating systems where OpenSSL is available.

Prepare the OpenSSL Configuration File

Create a text file called openssl.conf from your current working directory on the machine OpenSSL is installed. Copy and paste these lines to provide OpenSSL with the necessary configurations for certificate signing. You can edit this file to suit your needs.

```
[ ca ]
default_ca  = IntermCA

[ RootCA ]

dir     = ./RootCA
certs       = $dir/RootCA.db.certs
crl_dir = $dir/RootCA.db.crl
database    = $dir/RootCA.db.index
unique_subject = yes
new_certs_dir   = $dir/RootCA.db.certs
certificate = $dir/RootCA.crt
serial      = $dir/RootCA.db.serial
#crlnumber   = $dir/RootCA.db.crlserial
private_key = $dir/RootCA.key
RANDFILE    = $dir/RootCA.db.rand
name_opt    = ca_default
cert_opt    = ca_default
############################## Modify default days for certificates signed by Root CA (Intermediate cert
default_days    = 360
default_md  = sha256
preserve    = no
policy          = optional_policy

[ IntermCA ]

dir     = ./IntermCA
certs       = $dir/IntermCA.db.certs
crl_dir = $dir/IntermCA.db.crl
database    = $dir/IntermCA.db.index
unique_subject = yes
new_certs_dir   = $dir/IntermCA.db.certs
certificate = $dir/IntermCA.crt
serial      = $dir/IntermCA.db.serial
private_key = $dir/IntermCA.key
RANDFILE    = $dir/IntermCA.db.rand
name_opt    = ca_default
cert_opt    = ca_default
# Certificate field options
############################## Modify default days for certificates signed by Intermediate CA cert (devi
default_days    = 1000
#default_crl_days  = 1000
default_md  = sha256
# use public key default MD
preserve    = no
policy          = optional_policy


[ optional_policy ]
countryName     = optional
stateOrProvinceName = optional
localityName        = optional
organizationName    = optional
organizationalUnitName  = optional
commonName      = supplied
```

```
[ req ]
default_bits        = 2048
default_keyfile     = privkey.pem
distinguished_name  = req_distinguished_name
attributes      = req_attributes
x509_extensions = v3_ca # The extentions to add to the signed cert
string_mask = nombstr

[ req_distinguished_name ]
countryName         = Country Name
countryName_default     = MX
countryName_min         = 2
countryName_max         = 2

stateOrProvinceName     = State or province
stateOrProvinceName_default = CDMX

localityName            = Locality
localityName_default        = CDMX

organizationName      = Organization name
organizationName_default  = Cisco lab

organizationalUnitName      = Organizational unit
organizationalUnitName_default  = Cisco Wireless

commonName          = Common name
commonName_max          = 64



[ req_attributes ]
# challengePassword     = A challenge password
# challengePassword_min     = 4
# challengePassword_max     = 20

#This section contains the extensions used for the Intermediate CA certificate

[ v3_ca ]
# Extensions for a typical CA
basicConstraints = CA:true
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer:always
subjectAltName = @Intermediate_alt_names



[ v3_req ]
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth, clientAuth

[ crl_ext ]
# CRL extensions.
#authorityKeyIdentifier=keyid:always,issuer:always


#DEFINE HERE SANS/IPs NEEDED for Intermediate CA device certificates
[Intermediate_alt_names]
DNS.1   = Intermediate.example.com
DNS.2   = Intermediate2.example.com
```

```
#Section for endpoint certificate CSR generation
[ endpoint_req_ext ]
subjectAltName = _alt_names

#Section for endpoint certificate sign by CA
[ Endpoint ]
basicConstraints=CA:FALSE
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
#Change the key usage according to the certificate usage needs
extendedKeyUsage = clientAuth
subjectAltName = _alt_names

#Define here SANS/IPs needed for  Endpoint certificates
[endpoint_alt_names]
DNS.1   = Endpoint.example.com
DNS.2   = Endpoint2.example.com

#Section for IOS-XE device certificate CSR generation
[ device_req_ext ]
subjectAltName = @IOS_alt_names

#Section for IOS-XE certificate sign by CA
[ IOS_cert ]
basicConstraints=CA:FALSE
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
#Change the key usage according to the certificate usage needs
extendedKeyUsage = clientAuth , serverAuth
subjectAltName = @IOS_alt_names

#Define here SANS/IPs needed for IOS-XE certificates
[IOS_alt_names]
DNS.1   = IOSXE.example.com
DNS.2   = IOSXE2.example.com
```

Create Initial Files for the Certificate Authorities

Create a folder on the current directory called **RootCA**. Inside it, create 3 more folders called **RootCA.db.tmp**, **RootCA.db.certs**, and **RootCA.db.crl**.

```
mkdir RootCA
mkdir RootCA/RootCA.db.tmp
mkdir RootCA/RootCA.db.certs
mkdir RootCA/RootCA.db.crl
```

Create a file called **RootCA.db.serial** inside the **RootCA** folder. This file needs to contain the initial value for the certificates serial number, **01** is the value selected on this case.

Create a file called **RootCA.db.crlserial** inside the **RootCA** folder. This file needs to contain the initial value for the certificate revocation list number, **01** is the value selected on this case.

```
echo 01 > RootCA/RootCA.db.serial
echo 01 > RootCA/RootCA.db.crlserial
```

Create a file called **RootCA.db.index** inside the **RootCA** folder.

```
touch RootCA/RootCA.db.index
```

Create a file named **RootCA.db.rand** inside the **RootCA** folder and populate it with 8192 random bytes to serve as the seed of the internal random number generator.

```
openssl rand -out RootCA/RootCA.db.rand 8192
```

Create a folder on the current directory called **IntermCA**. Inside it, create 3 more folders called **IntermCA.db.tmp**, **IntermCA.db.certs**, and **IntermCA.db.crl**.

```
mkdir IntermCA
mkdir IntermCA/IntermCA.db.tmp
mkdir IntermCA/IntermCA.db.certs
mkdir IntermCA/IntermCA.db.crl
```

Create a file called **IntermCA.db.serial** inside the **IntermCA** folder. This file needs to contain the initial value for the certificates serial number, **01** is the value selected on this case.

Create a file called **IntermCA.db.crlserial** inside the **IntermCA** folder. This file needs to contain the initial value for the certificate revocation list number, **01** is the value selected on this case.

```
echo 01 > IntermCA/IntermCA.db.serial
echo 01 > IntermCA/IntermCA.db.crlserial
```

Create a file named **IntermCA.db.index** inside the **IntermCA** folder.

Create a file named **IntermCA.db.rand** inside the **IntermCA** folder and populate it with 8192 random bytes to serve as the seed of the internal random number generator.

```
touch IntermCA/IntermCA.db.index
```

Create a file named **IntermCA.db.rand** inside the **IntermCA** folder and populate it with 8192 random bytes to serve as the seed of the internal random number generator.

```
openssl rand -out IntermCA/IntermCA.db.rand 8192
```

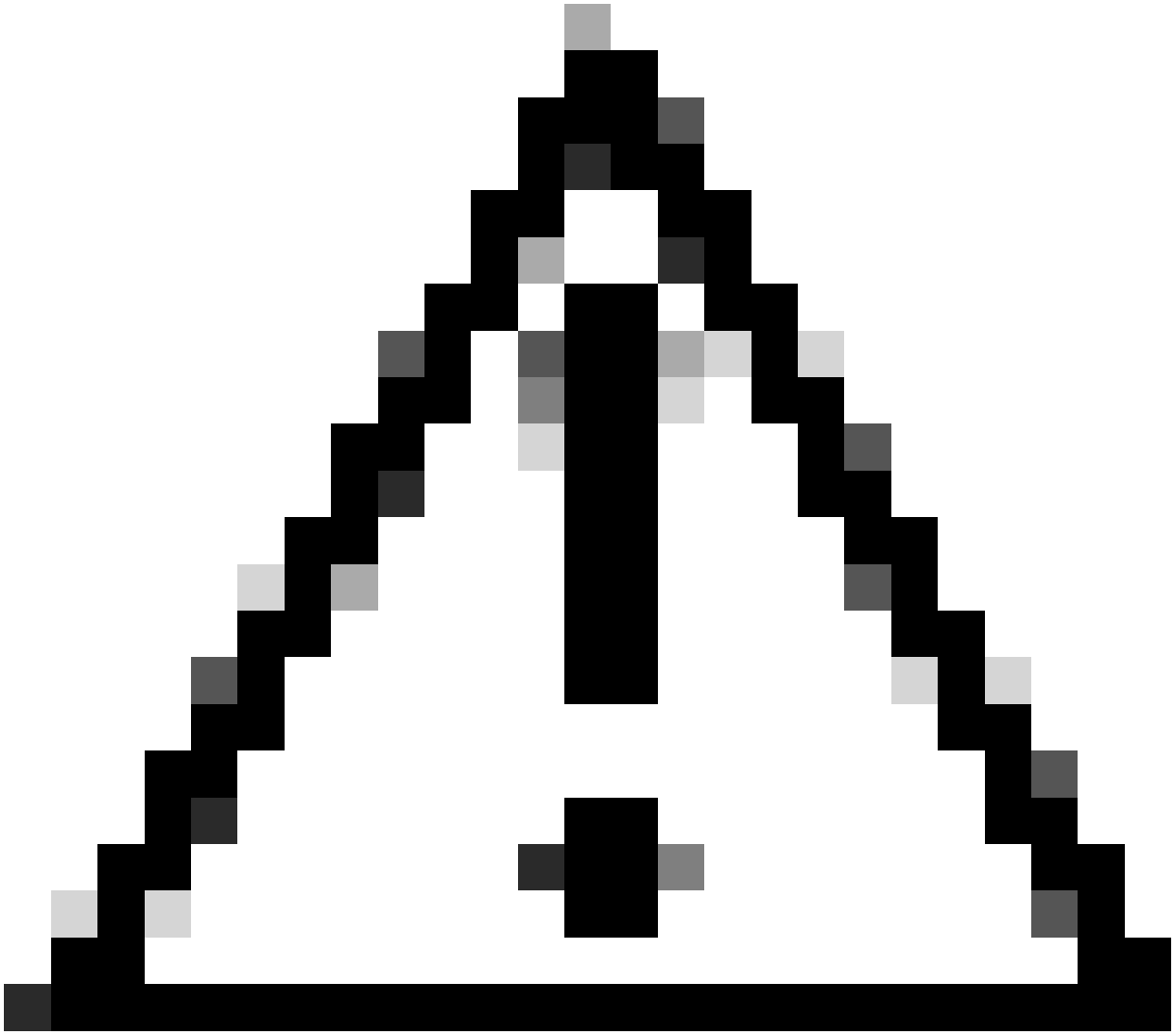
This is the file structure after the creation of all initial Root and Intermediate CA files.

```
mariomed@CSCO-W-PF320YP6:/mnt/c/Users/mariomed/radsecfiles1$ tree
.
├── IntermCA
│   ├── IntermCA.db.certs
│   ├── IntermCA.db.crl
│   ├── IntermCA.db.crlserial
│   ├── IntermCA.db.index
│   ├── IntermCA.db.rand
│   ├── IntermCA.db.serial
│   └── IntermCA.db.tmp
├── RootCA
│   ├── RootCA.db.certs
│   ├── RootCA.db.crl
│   ├── RootCA.db.crlserial
│   ├── RootCA.db.index
│   ├── RootCA.db.rand
│   ├── RootCA.db.serial
│   └── RootCA.db.tmp
└── openssl.cnf
```

Create Root CA Certificate

Run this command to create the private key for the Root CA.

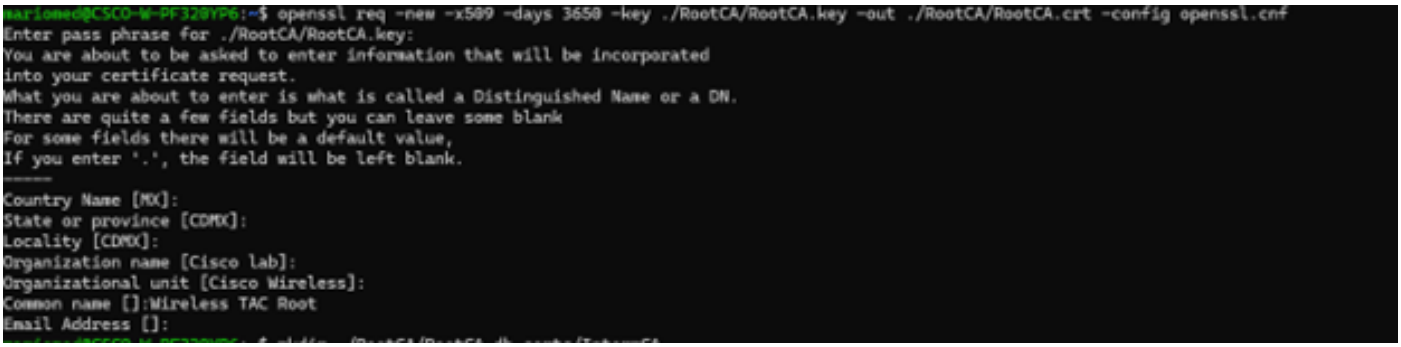
```
openssl genrsa -des3 -out ./RootCA/RootCA.key 4096
```



Caution: OpenSSL requires you to provide a passphrase when a key is generated. Keep the passphrase secret and the generated private key on a secure location. Anyone with access to it can issue certificates as your Root CA.

Create the root CA self signed certificate using the req command on openSSL. The -x509 flag internally creates a certificate signing request (CSR) and automatically self-signs it. Edit the -days parameter and subject alternative name. The perminal prompts you to provide a common name. Ensure the common name you enter matches the Subject Alternative Name (SAN).

```
openssl req -new -key ./RootCA/RootCA.key -out ./RootCA/RootCA.crt -config openssl.cnf -x509 -days 3650
```



OpenSSL Distinguished Name Interactive Prompt

The generated file is called RootCA.crt and is located inside the **RootCA** folder. This file is the Root CA certificate.

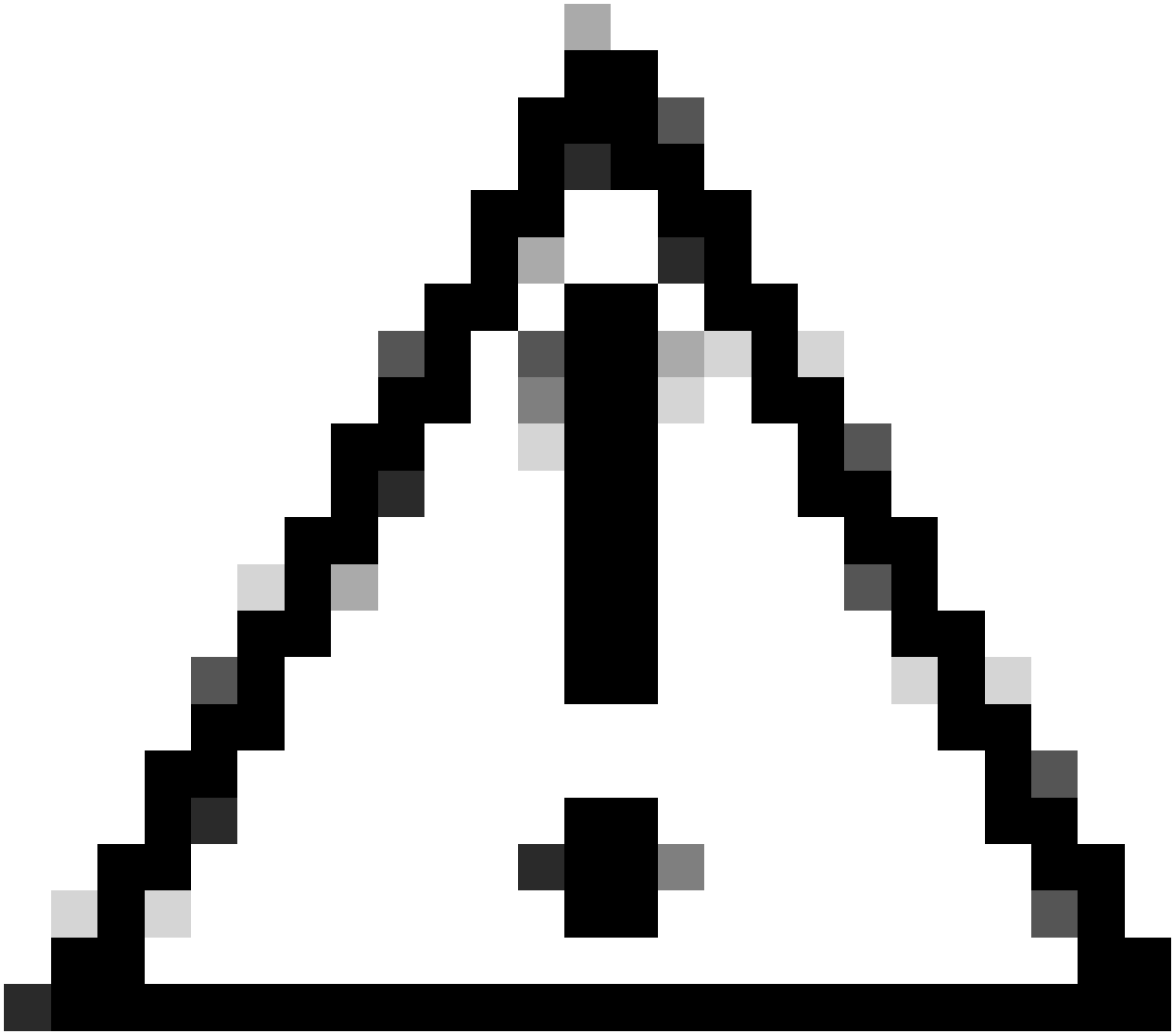
Create Intermediate CA Certificate

Create folder to store the signed Intermediate CA certificate inside the root folder.

```
mkdir ./RootCA/RootCA.db.certs/IntermCA
```

Create private key for intermediate certificate.

```
openssl genrsa -des3 -out ./RootCA/RootCA.db.certs/IntermCA/IntermCA.key 4096
```



Caution: OpenSSL requires you to provide a passphrase when a key is generated. Keep the passphrase secret and the generated private key on a secure location. Anyone with access to it can issue certificates as your Intermediate CA.

Create intermediate CA Certificate Signing Request. The terminal prompts you to enter the certificate information.

```
openssl req -new -key ./RootCA/RootCA.db.certs/IntermCA/IntermCA.key -out ./RootCA/RootCA.db.certs/Inte
```

Sign Intermediate CSR with the **RootCA** section of the openssl.cnf file.

```
openssl ca -config openssl.cnf -name RootCA -extensions v3_ca -out ./RootCA/RootCA.db.certs/IntermCA/In
```

The generated file is called IntermCA.crt and is located inside the RootCA folder. This file is the Root CA certificate.

Move the intermediate certificate and key to its own folder that you created as part of the initial files for the intermediate CA.

```
cp ./RootCA/RootCA.db.certs/IntermCA/IntermCA.crt ./RootCA/RootCA.db.certs/IntermCA/IntermCA.key ./Inte
```

This is the file structure after the creation of the private key and certificates for both initial Root and Intermediate CAs.

```
mariomed@CSCO-W-PF320YP6:/mnt/c/Users/mariomed/radsecfiles$ tree
.
├── IntermCA
│   ├── IntermCA.crt  <------Intermediate CA certficate
│   ├── IntermCA.db.certs
│   ├── IntermCA.db.crl
│   ├── IntermCA.db.crlserial
│   ├── IntermCA.db.index
│   ├── IntermCA.db.rand
│   ├── IntermCA.db.serial
│   ├── IntermCA.db.tmp
│   └── IntermCA.key  <------Intermediate CA private key
├── RootCA
│   ├── RootCA.crt    <------Root CA certficate
│   ├── RootCA.db.certs
│   │   ├── 01.pem
│   │   └── IntermCA
│   │       ├── IntermCA.crt
│   │       ├── IntermCA.csr
│   │       └── IntermCA.key
│   ├── RootCA.db.crl
│   ├── RootCA.db.crlserial
│   ├── RootCA.db.index
│   ├── RootCA.db.index.attr
│   ├── RootCA.db.index.old
│   ├── RootCA.db.rand
│   ├── RootCA.db.serial
│   ├── RootCA.db.serial.old
│   ├── RootCA.db.tmp
│   └── RootCA.key    <------Root CA private key
└── openssl.cnf
```

Create Device Certificates

Create Cisco IOS XE Device Certificate

Create a new folder to store the Cisco IOS XE device certificates.

```
mkdir ./IntermCA/IntermCA.db.certs/IOSdevice
```

Create the device private key **IOSdevice.key** and device CSR **IOSdevice.csr.** Use section **device_req_ext** to add the SANs under said section onto the CSR.

```
openssl req -newkey rsa:4096 -sha256 -keyout ./IntermCA/IntermCA.db.certs/IOSdevice/IOSdevice.key -node
```

Modify the opnessl.cnf file **[IOS_alt_names]** section so that the common name you provide on the CSR matches the SAN.

```
#Define here SANS/IPs needed for IOS-XE certificates
[IOS_alt_names]
DNS.1   = IOSXE.example.com
DNS.2   = IOSXE2.example.com
```

Sign IOS XE device CSR with intermediate CA IntermCA section. Use -config to point to the openSSL configuration file and -extensions to point to the IOS_cert section. This keeps the SAN on the signed certificate.

```
openssl ca -config openssl.cnf -extensions IOS_cert -name IntermCA -out ./IntermCA/IntermCA.db.certs/IO
```

After this step, you have created a valid certificate for the IOS XE device called IOSdevice.crt with matching private key IOSdevice.key.

Optional - Create Endpoint Certificate

At this point, you have deployed a local CA and issued one certificate for your IOS XE device. You can also use this CA to generate endpoint identity certificates. These certificates are valid too, for example, perform Local EAP authentication on 9800 Wireless LAN controllers or even dot1x authentication with RADIUS servers. This section helps you to generate an endpoint certificate.

Create a folder to store the endpoint certficates.

```
mkdir ./IntermCA/IntermCA.db.certs/Endpoint
```

Modify the openSSL.cnf file **[endpoint_alt_names]** section so that the common name you provide on the CSR matches the SAN.

```
#Define here SANS/IPs needed for  Endpoint certificates
[endpoint_alt_names]
```

```
DNS.1   = Endpoint.example.com
DNS.2   = Endpoint2.example.com
```

Create the endpoint private key and WLC CSR with the use of section endpoint_req_ext for SANs.

```
openssl req -newkey rsa:2048 -keyout ./IntermCA/IntermCA.db.certs/Endpoint/Endpoint.key -nodes -config
```

Sign the Endpoint device certificate.

```
openssl ca -config openssl.cnf -extensions Endpoint -name IntermCA -out ./IntermCA/IntermCA.db.certs/En
```

Import Certificate to the Cisco IOS XE device

Create a file which contains the Root CA and Intermediate CA on the same file and save it to **./IntermCA/IntermCA.db.certs/WLC/** folder with name **certfile.crt** as is required for import to the Cisco IOS XE device.

```
cat ./RootCA/RootCA.crt ./IntermCA/IntermCA.crt > ./IntermCA/IntermCA.db.certs/IOSdevice/certfile.crt
```

The 9800 series WLC uses different commands to create the pfx file for certificate import. To create your pfx file, run one of these commands according to the Cisco IOS XE version.

Refer to Generate and Download CSR Certificates on Catalyst 9800 WLCs for detailed information about the certificate import process

For versions older than 17.12.1:

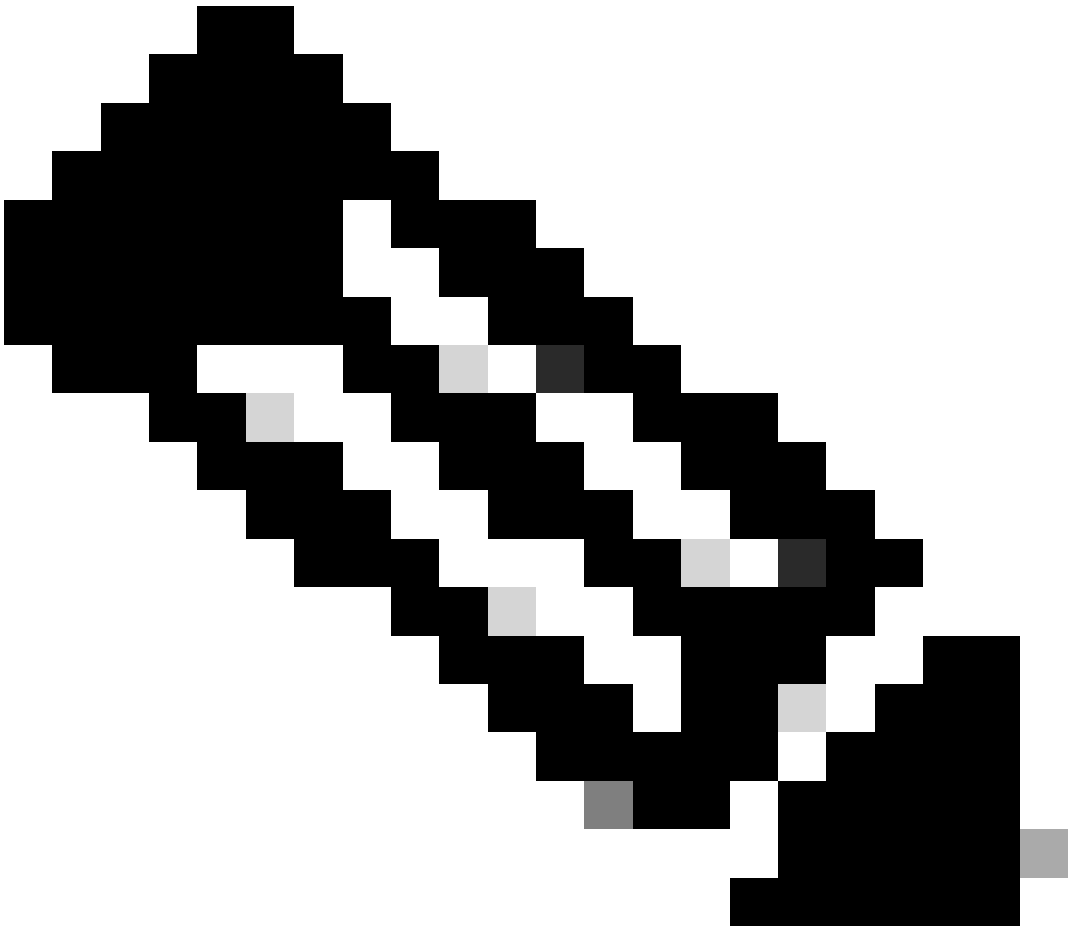
```
openssl pkcs12 -export -macalg sha1 -legacy -descert -out ./IntermCA/IntermCA.db.certs/IOSdevice/IOSdev
```

For version 17.12.1 or later:

```
openssl pkcs12 -export -out ./IntermCA/IntermCA.db.certs/IOSdevice/IOSdevice.pfx -inkey ./IntermCA/Inte
```

Import the **IOSdevice.pfx** certificate to the Cisco IOS XE device:

```
WLC# configure terminal
WLC(config)#crypto pki import <trustpoint-name> pkcs12 [tftp://<TFTP-IP>/<cert-filename> | ftp://<FTP-I
```



> **Note**: Ensure the CA certificates created for this guide are trusted by the devices that need to verify the device certificate. For example, If the device certificate is used for web admin purposes on the Cisco IOS XE device, any computer or browser accessing the admin portal needs to have the CA certificates on its trust store.

Disable revocation check for the certificates as there is no online certificate revocation list the Cisco IOS XE device can check from the CA you have deployed.
You must disable it on all the trustpoints that are part of the verification path. The root CA trustpoint has the same name as the Intermediate/Device trustpoint with the string -rrr1 appended at the end.

```
9800#configure terminal

9800(config)#crypto pki trustpoint IOSdevice.pfx
9800(config)#revocation-check none
9800(config)#exit

9800(config)#crypto pki trustpoint IOSdevice.pfx-rrr1
9800(config)#revocation-check none
9800(config)#exit
```

Verify

Verify Certificate Information on OpenSSL

To verify the certificate information for the created certificates, on the Linux terminal run the command:

```
openssl x509 -in <path to cert> -text -noout
```

It shows the full certificate information.

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 2 (0x2)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = MX, ST = CDMX, L = CDMX, O = Cisco lab, OU = Cisco Wireless, CN = Intermediate.example.com
        Validity
            Not Before: Jul 18 19:14:57 2024 GMT
            Not After : Apr 14 19:14:57 2027 GMT
        Subject: C = MX, ST = CDMX, L = CDMX, O = Cisco lab, OU = Cisco Wireless, CN = WLC.example.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:b1:10:7d:6c:6c:14:2f:18:a6:0b:69:d9:60:03:
                    56:2d:48:22:f0:42:10:65:44:24:3b:54:e1:4b:87:
                    b8:ab:c5:5f:f6:a1:a3:5e:f6:3c:c5:45:cc:01:6d:
                    df:e8:a7:81:28:50:44:54:4c:af:a0:56:cf:06:be:
                    10:7e:e2:46:42:ea:3c:b9:d4:03:75:08:84:70:36:
                    bb:3d:95:3b:e2:86:e6:f7:d9:4d:00:28:c4:3c:cb:
                    f8:6d:37:5c:89:28:c1:75:b1:7e:fa:bd:91:cf:8e:
                    5c:a2:37:4f:71:da:6a:04:ee:ba:68:bf:4d:f2:d3:
                    ae:aa:13:42:3b:ff:a0:b3:65:c9:ff:f6:9a:06:d7:
                    6c:08:10:e0:b9:d8:ca:93:2d:e5:5d:7b:74:cd:93:
                    68:b1:46:c7:35:d7:6b:0f:a6:ae:34:e6:23:d1:c8:
                    d3:bf:c0:85:ab:2d:02:a8:dd:54:77:e3:32:61:4e:
                    33:58:b0:62:12:82:42:ae:2b:69:f0:5f:0c:90:c7:
                    9c:ef:b9:9c:fc:29:e2:2c:cb:b4:a9:01:fa:5d:3c:
                    97:11:67:cc:25:96:01:3d:26:1a:43:34:bd:43:b0:
                    a0:f1:ec:a0:c7:98:ad:32:32:99:9c:6b:61:af:57:
                    53:ee:20:cc:d5:ed:db:1c:5c:65:51:42:8c:28:bf:
                    62:bf
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Subject Key Identifier:
                87:89:CA:28:06:95:D5:CE:7C:66:B4:75:81:AA:D4:19:EC:43:01:BB
            X509v3 Authority Key Identifier:
                keyid:2B:08:D8:4C:23:72:5B:62:03:EA:44:F6:9E:D9:F7:75:2E:64:97:DE
                DirName:/C=MX/ST=CDMX/L=CDMX/O=Cisco lab/OU=Cisco Wireless/CN=RootCA
                serial:01
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
            X509v3 Subject Alternative Name:
                DNS:WLC.example.com, DNS:WLC2.example.com
    Signature Algorithm: sha256WithRSAEncryption
    Signature Value:
```

Cisco IOS XE Device Certificate Information as Shown by OpenSSL

Verify Certificate Information on the Cisco IOS XE Device.

Command show crypto pki certificates verbose prints the certificate information of all available certificates on the device.

```
9800#show crypto pki certificates verbose
CA Certificate <------Type of certificate
  Status: Available
  Version: 3
  Certificate Serial Number (hex): 2A352E27C69021ECE1AA61751CA1F233E0636FB1
  Certificate Usage: General Purpose
  Issuer: <-------------DN for issuer
    cn=RootCA
    ou=Cisco Wireless
    o=Cisco lab
    l=CDMX
    st=CDMX
    c=MX
```

```
  Subject: <--------------DN for subject
    cn=RootCA
    ou=Cisco Wireless
    o=Cisco lab
    l=CDMX
    st=CDMX
    c=MX
  Validity Date:  <--------Validity date
    start date: 14:54:02 Central Jul 22 2024
    end   date: 14:54:02 Central Jul 20 2034
  Subject Key Info:
    Public Key Algorithm: rsaEncryption
    RSA Public Key: (2048 bit)  <--------------Key size
  Signature Algorithm: SHA256 with RSA Encryption
  Fingerprint MD5: 432021B5 B4BE15F5 A537385C 4FAB9A94
  Fingerprint SHA1: 86D18427 BE619A2A 6C20C314 9EDAAEB2 6B4DFE87
  X509v3 extensions:
    X509v3 Subject Key ID: 57DEEBD8 3214CA05 176F0CD6 6C842EBC 9ABFF7D8
    X509v3 Basic Constraints:
        CA: TRUE
    X509v3 Subject Alternative Name:
        RootCA  <------------SANs
        IP Address :
        OtherNames :
    X509v3 Authority Key ID: 57DEEBD8 3214CA05 176F0CD6 6C842EBC 9ABFF7D8
    Authority Info Access:
  Cert install time: 16:42:09 Central Jul 22 2024
  Associated Trustpoints: WLC.pfx-rrr1  <----------Associated trustpoint
  Storage: nvram:RootCA#6FB1CA.cer
```

Troubleshoot

Revocation Check is in Place

When the certificates are imported to Cisco IOS XE, the newly created trustpoints have revocation check enabled. If a certificate is presented to the device that needs to use the imported certificate trustpoints for validation, the device searches for a non-existent Certificate Revocation List and fails. The message is printed on the terminal.

```
Jul 17 21:50:39.068: %PKI-3-CRL_FETCH_FAIL: CRL fetch for trustpoint WLC1.pfx failed
                     Reason : Enrollment URL not configured.
```

Ensure each trustpoint in the verification path for the certificates contains the command revocation-check none.

Related Information

- Generate and Download CSR Certificates on Catalyst 9800 WLCs
- Configure CA Signed Certificates with IOS XE PKI
- Security and VPN Configuration Guide, Cisco IOS XE 17.x
- Understand Certificate Information to Create a Chain for 9800 WLC