Cisco Meeting Server

Cisco Meeting Server Release 2.9

Certificate Guidelines for Scalable and Resilient Server Deployments

December 01, 2020

Contents

Change History

Date	Change Summary
December 01, 2020	Added note to section: CSR for the Web Bridge
June 02, 2020	Minor corrections.
May 16, 2020	Updated for Cisco Meeting Server version 2.9
September 30, 2019	Minor correction.
September 27, 2019	Minor correction.
August 12, 2019	Released for Meeting Server version 2.7.
February 5, 2019	Correction to pki csr example command.
January 21, 2019	Invalid example csr name corrected.
January 02, 2019	Changed title to "...2.4 and later", no changes for version 2.5.
December 14, 2018	Minor correction.
November 29, 2018	Minor correction.
October 02, 2018	Updated for Cisco Meeting Server 2.4. Introduction of XMPP certificate validation and Call Bridge cluster validation. Announced removal of H.323 Gateway, SIP Edge, TURN Server, XMPP Server and Load Balancer components in future version of Meeting Server software.
January 12, 2018	Released for Cisco Meeting Server 2.3
May 10, 2017	Minor corrections. Recommended using Cisco Expressway for Federated Lync/S4B. Released for 2.2.
December 19, 2016	Minor corrections and released for 2.1
August, 03, 2016	Rebranded for Cisco Meeting Server 2.0.

1 Introduction

The Cisco Meeting Server software can be hosted on specific servers based on Cisco Unified Computing Server (UCS) technology as well as on the X-Series hardware, or on a specification-based VM server. Cisco Meeting Server is referred to as the Meeting Server throughout this document.

The Cisco Meeting Server is very secure, most of the services and applications running on the server use the TLS cryptyographic protocol for communication. TLS allows communicating parties to exchange X.509 certificates and public keys in order authenticate the other party, and exchange encryption algorithms to encrypt data transmitted between the parties.

This Certificate Guidelines document explains how to create and install certificates for a Scalable and Resilient deployment.

Note: The Cisco Meeting Server software is referred to as the Meeting Server throughout the remainder of this guide.

1.1 How to use this guide

The remainder of this chapter explains concepts that you will need to understand in order to deploy certificates across the Meeting Server deployment. Skip this if you are already familiar with PKI, certificates, and trust stores.

Chapter details where certificates are required within the scalable and resilient server model, and the type of certificates required.

Chapter explains how to create certificates.

Chapter covers installing the certificates on the Meeting Servers.

Chapter 5 provides troubleshooting information for typical certificate related issues.

Chapter explains how you can quickly create self-signed certificates.

Appendix A covers using OpenSSL rather than the Meeting Server pki command if you prefer to use OpenSSL.

Appendix B provides an overview of permitted filename extensions for certificate files and private keys.

Appendix C lists the MMP pki commands.

Appendix D provides certificate and configuration information if you are deploying Web Bridge3 to use Cisco Meeting Server web app.

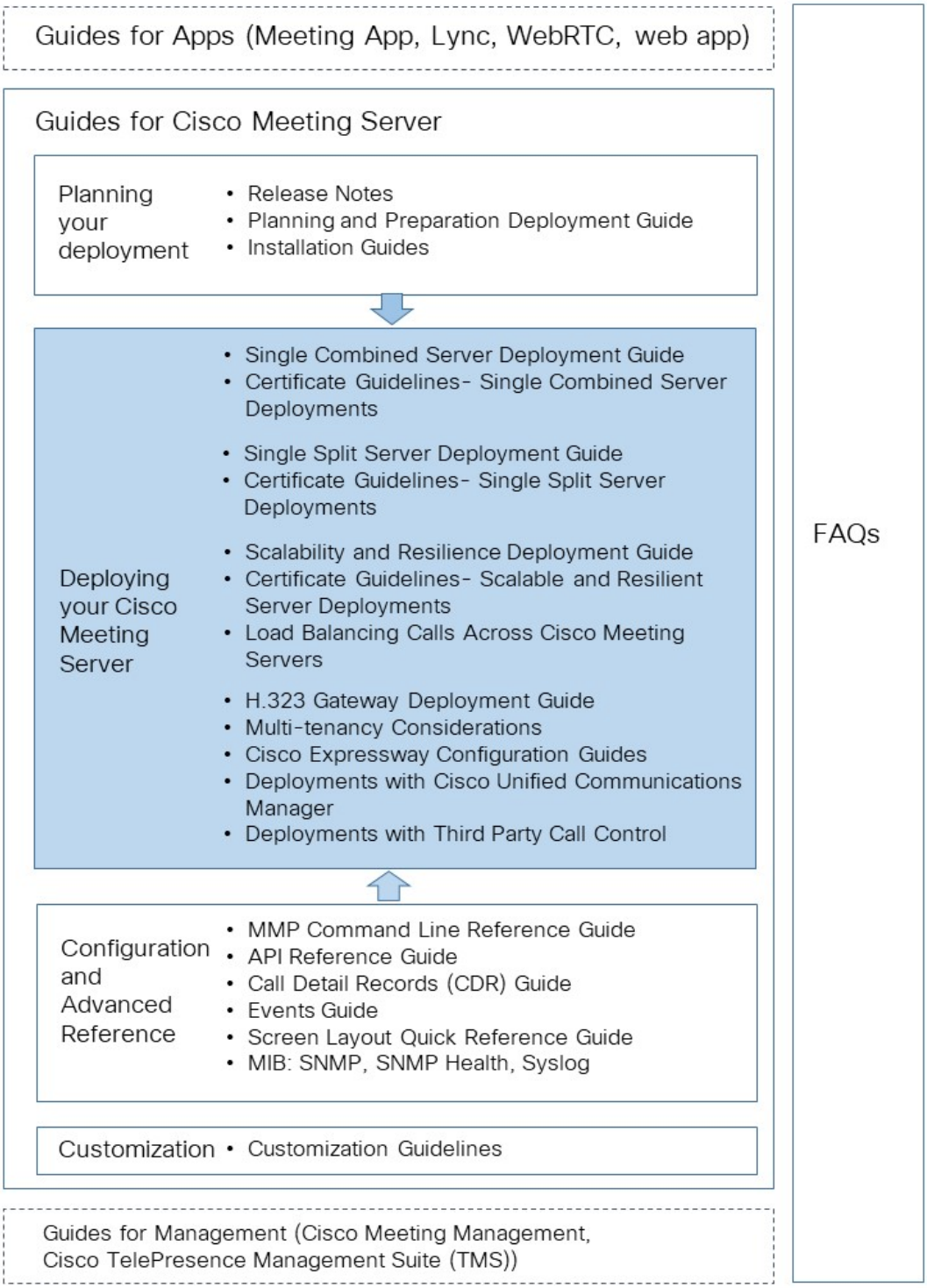
In version 2.9, Meeting Server introduces the new Cisco Meeting Server web app which is a browser-based client for Cisco Meeting Server that lets users join meetings (audio and video). To use this feature you need to deploy the new Web Bridge 3. In addition, Meeting Server

version 2.9 still offers the original Cisco Meeting App WebRTC (also referred to here as Web Bridge 2).

Note: Web app does not require XMPP. The XMPP component will be removed from a future version. Cisco Meeting App WebRTC still requires XMPP.

This guide is part of the documentation set (shown in Figure 1) for the Meeting Server.

Figure 1: Overview of guides covering the Cisco Meeting Server



These documents can be found on cisco.com.

1.2 Brief overview of PKI

Public key infrastructure (PKI) provides a mechanism to secure communications and validate identities of communicating parties. Communications are made secure through encryption, and identities are validated through the use of public/private key pairs and digital identity certificates.

1.2.1 Public/private key pair

A public and private key pair comprises two uniquely related cryptographic keys mathematically related. Whatever is encrypted with a public key may only be decrypted by its corresponding private key (which must be kept secret), and vice versa.

1.2.2 Certificates

A certificate is a wrapper around the public key, and provides information about the owner of the public key. It typically contains the name of the entity to which the certificate is issued, contact details for the owner, validity dates (when the certificate is valid), and issuer (the authority that issued the certificate). Certificates need to be signed by trustworthy authorities that can validate that the owner is who they claim to be. Certificate Authorities (CAs) are trustworthy authorities that certify the identities of individuals, organizations, and computers on the network.

When an entity requires a certificate, it first generates a public/private key pair. It then creates a Certificate Signing Request (.csr) file which contains the entity's public key and information identifying the entity (see Table 1). The entity signs the .csr file using their private key and sends the .csr file to a CA for processing. Depending on the level of verification required, the entity may send the .csr file to either a public CA, such as Verisign, or use an internal CA, for example Active Directory server with the Active Directory Certificate Services Role installed.

The CA uses the .csr file and public key to verify the identity of the entity. If verification is successful, the CA issues a digital identity certificate to the entity, which is proof that the entity named in the certificate is the owner of the public key and private key set. The digital identity certificate is used by the entity to give other entities on the network a high level of assurance that the public key really belongs to the owner of the private key.

Table 1: Information in a .csr file

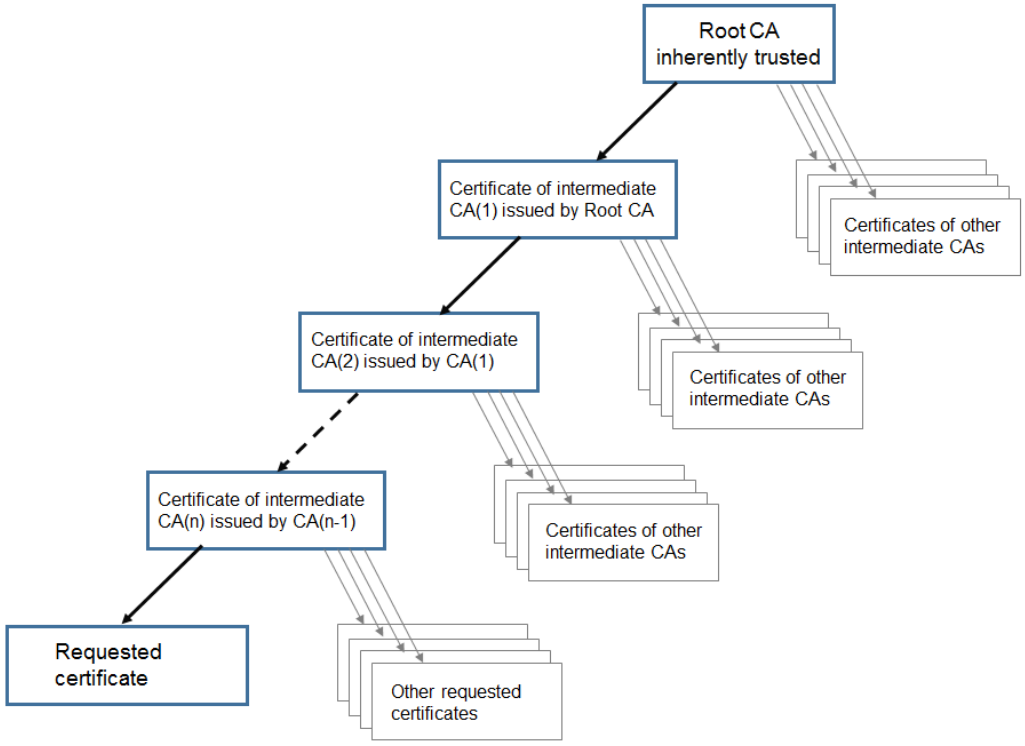
Information	Description
Common Name (CN)	This is the fully qualified domain name that you wish to secure e.g. 'www.example.com'.
Organization or Business name (O)	Usually the legal incorporated name of a company. It should include any suffixes such as Ltd., Inc., or Corp.
Organizational unit or Department name (OU)	For example, Support, IT, Engineering, Finance.

Information	Description
Location (L)	City or town. For example, London, Boston, Milan, Berlin.
Province, Region, County or State (ST)	For example, Buckinghamshire, New Jersey. Do not abbreviate.
Country (C)	The two-letter ISO code for the country where your organization is located. For example, US, GB, FR.
An email address	An email address to contact the organization. Usually the email address of the certificate administrator or IT department.
Subject Alternative Name (subjectAltName)	From X509 Version 3 (RFC 2459), SSL certificates are allowed to specify multiple names that the certificate should match. subjectAltName (SAN) can contain, for example, email addresses, IP addresses, regular DNS host names.

1.2.3 Chain of Trust

When an entity is challenged by another to provide its certificate for authentication, that entity needs to present its own certificate, along with a series of other certificates that establish a link to a Certificate Authority that the challenging party trusts (usually known as the Root Certificate Authority). This hierarchy of certificates, linking an entity's certificate to a Root CA, is called a 'chain of trust'. It is quite often the case that a Root CA has signed a certificate for another Certificate Authority (known as an Intermediate CA), which in turn has signed the entity's certificate. In that case, the entity needs to present both its own certificate and the certificate for this Intermediate CA that has been issued by the Root CA. If the entity only presented its own certificate, without establishing a link to a trusted Root CA, the challenging party will not trust the certificate presented. The series of certificates that link an entity's certificate to a root CA is known as 'intermediate certificates' as they are issued to Intermediate CAs.

Figure 2: Certificate Chain of Trust



To enable connecting devices to verify a chain of trust, every certificate includes two fields: "Issued To" and "Issued By". An intermediate CA will show different information in these two fields, to show a connecting device where to continue checking, if necessary, in order to establish trust. Root CA certificates are "Issued To" and "Issued By" themselves, so no further checking is possible.

For example, if Entity A (web server www.example.com) is challenged for authentication by Entity B (web client), Entity A will need to present its certificate and certificate chain to Entity B.

Figure 3: Certificate chain for Entity A

Certificate 1 – Issued To: example.com; Issued By: Intermediate CA 1
Certificate 2 – Issued To: Intermediate CA 1; Issued By: Intermediate CA 2
Certificate 3 – Issued To: Intermediate CA 2; Issued By: Root CA

Providing Entity B has in its trust store the certificate for Root CA, a secure connection can be established between Entity A and Entity B. Entity B can use Entity A's public key to encrypt messages and send them to Entity A. Only Entity A has access to the private key, so only Entity A can decrypt the messages.

Note: This process is called "certificate chaining" and intermediate CA certificates are sometimes called "chained certificates".

CAUTION: The database nodes forming the cluster must be configured with a trusted root CA certificate so that only legitimate nodes can connect to the cluster. The nodes will trust connections that present a certificate chain that ends with a trusted root certificate. Therefore each database cluster must use a dedicated root certificate, the root certificate or intermediate certificates must not be used for any other purpose.

1.2.4 Certificate bundles

A certificate bundle is a single file (with an extension of .pem, .cer or.crt) holding a copy of the Root CA's certificate and all intermediate certificates in the chain. The certificates need to be in sequence with the certificate of the Root CA being last in the certificate bundle. External clients (for example web browsers and XMPP clients) require the certificate and certificate bundle to be presented by the Web Bridge 2 and XMPP server respectively, when setting up a secure connection. If Call Bridge establishes a TLS trunk to a SIP peer, then Call Bridge will need to presents its certificate and certificate bundle to the SIP endpoint.

You can create a certificate bundle by using a plain text editor such as notepad. All of the characters including the -----BEGIN CERTIFICATE----- and -----END CERTIFICATE----- tags need to be inserted into the document. There should be no space between the certificates, for example no spaces or extra lines between -----END CERTIFICATE----- of certificate 1 and -----BEGIN CERTIFICATE----- of certificate 2. Certificate 1 will end with -----END CERTIFICATE----- and the very next line will have -----BEGIN CERTIFICATE---- for certificate 2. At the end of the file there should be 1 extra line. Save the file with an extension of .pem, .cer, or .crt.

Note: Web Bridge 3 requires all of its certificate definitions to use a bundle of certificates, i.e. a full chain file. This is different to the certificate implementation for Web Bridge 2.

1.2.5 Trust stores

Web browsers and other clients hold a list of signing authorities that they trust and therefore, by a "chain of trust", the servers they can trust. These trusted CAs are held in a 'trust store' on the client. When the trusted CA issues a revocation list, the client updates its trust store removing the entities in the revocation list from the store.

For a connecting client (or device) to trust a certificate, the client will check whether the CA of the certificate is held in the client's trust store. If the certificate was not issued by a trusted CA, the connecting client will then check to see if the certificate of the issuing CA was issued by a trusted CA, this will be repeated up the chain until either a trusted CA is found or no trusted CA can be found. If a trusted CA is found, a secure connection will be established between the

client and the server. If a trusted CA cannot be found, then the connecting client will usually display an error message.

2 Certificates required for the deployment

This chapter explains where certificates are required to establish secure connections in the scalable and resilient server deployment, and the type of certificates required.

Note about removing components from Cisco Meeting Server software: The following components will be removed from a future version of the software: H.323 Gateway, SIP Edge, TURN Server, XMPP Server and Load Balancer. You are advised to migrate your deployment over to using the Cisco Expressway as your edge device and to start using the WebRTC app rather than the Cisco Meeting App thick client.

In version 2.9, Meeting Server introduces the new Cisco Meeting Server web app which is a browser-based client for Cisco Meeting Server that lets users join meetings (audio and video). To use this feature you need to deploy the new Web Bridge 3. In addition, Meeting Server version 2.9 still offers the original Cisco Meeting App WebRTC (also referred to here as Web Bridge 2).

For more information on Web Bridge 3 certificate requirements, see Appendix D.

Note: Web app does not require XMPP. The XMPP component will be removed from a future version. Cisco Meeting App WebRTC still requires XMPP.

2.1 Using Cisco Expressway as the edge for the Meeting Server

Over the previous few software releases for Cisco Expressway and Meeting Server, edge features have been developed to enable the Expressway to be used as the edge device in Meeting Server deployments. From version 2.4, you should start migrating your Meeting Server deployments from using the Meeting Server SIP edge component (SIP and Lync Call Traversal feature) and the Meeting Server TURN server, to using the Expressway X8.11 TURN server. For details on configuring the Cisco Expressway as the Meeting Server edge device see Cisco Expressway Traffic Classification Deployment guide.

As the edge device for the Meeting Server, the Expressway provides:

- edge support for Microsoft clients on Lync or Skype for Business infrastructure in other organizations,or Skype for Business clients on Office 365 (not "consumer" versions of Skype) connecting to Meeting Server dual homed conferences,
- support for standards based SIP endpoints connecting to Meeting Server hosted conferences,
- edge support for off-premise Cisco Meeting WebRTC App (thin client) using TCP port 443.

Note: The Expressway does not support web proxy of XMPP traffic. Remote Cisco Meeting App thick clients (Windows/Mac desktop or iOS) need to connect to the Load Balancer component of the Meeting Server for XMPP traffic.

Note: In deployments involving on-premise Microsoft infrastructure and the Meeting Server, the Meeting Server must use the Microsoft Edge server to traverse Microsoft calls into and out of the organization.

2.1.1 Expressway Reverse Web Proxy

From X8.10, Expressway includes a Reverse Web Proxy to connect the Cisco Meeting WebRTC App to the Web Bridge of the Meeting Server. You can use this reverse web to avoid the Web Bridge directly connecting to the internet. You need to enable the Web Proxy through the Mobile and Remote Access mode on the Expressway-C and the Expressway-E, but you do not need to completely configure MRA. For information on using the Web Proxy, see the "Cisco Expressway Web Proxy for Cisco Meeting Server Deployment Guide" available under the X8.10 listing here.

Connecting the WebRTC App also needs TURN media relays. Currently, you can use either the TURN server on Expressway-E or the TURN server component on the Meeting Server.

Note: Expressway cannot proxy the XMPP signaling required for the Cisco Meeting App thick clients (desktop and iOS).

2.2 Public or Internal CA signed certificates

Applications on the Meeting Server that interface to external devices, need to be trusted by the external devices, and require certificates signed by a public CA. Applications that interface internally within the Meeting Server only require certificates signed by an internal CA. Internal CA signed certificates can be generated by a local or organizational Certificate Authority, such as an Active Directory server with the Active Directory Certificate Services Role installed, see Section 3.3.

The applications that require public CA signed certificates are shown in Table 2. Applications that only require internal CA signed certificates are shown in Table 3.

For information on using wildcard certificates on the Meeting Server and other certificate related FAQs, go to this link.

Note: In deployments where the Meeting Server is trunked to Cisco Unified Communications Manager, Cisco Unified Communications Manager will accept the Call Bridge certificate signed by a public or internal Certificate Authority; but Cisco Unified Communications Manager requires the Call Bridge certificate to be signed using a template that allows for an Extended Key Usage containing both TLS Web Client Authentication and TLS Web Server Authentication.

Table 2: Public CA signed certificates (scalable and resilient server model)

Applications requiring public CA signed certificate	Reason
Web Bridge 2 (only if WebRTC Apps are used)	The Cisco Meeting WebRTC App requires a public CA signed certificate from the Web Bridge in order to trust the connection. Note: If using the Expressway Reverse Web Proxy to connect WebRTC apps to the Web Bridge, the Expressway-E certificate must have an entry in the SAN field which matches the hostname (URI) of the Web Bridge. See Section 2.1.1.
XMPP server (only if native Cisco Meeting Apps are used)	Native Cisco Meeting App require a public CA signed certificate from the XMPP server in order to trust the connection. From version 2.4, the Call Bridge and Web Bridge trust stores can hold a certificate allowed list for XMPP server verification, see Section 4.11.
Call Bridge (only if Meeting Server connected on a public network for direct Lync federation)	Lync Edge server requires a public CA signed certificate from the Call Bridge if doing direct federation.
TURN server	If you configure TLS on your TURN server, then the TURN server will require a certificate/key pair similar to that created for the Web Bridge, so that the WebRTC client trusts the connection. The certificate should be signed by the same Certificate Authority as used for the Web Bridge certificate

Table 3: Internal CA signed certificates (scalable and resilient server model)

Applications that can use internal CA signed certificate	Reason
Web Admin	The Meeting Server only allows HTTPS connection to the interface of the Web Admin, so a certificate is required for the Web Admin. Note: The Meeting Server API is routed through the interface of the Web Admin, so a certificate is required even if you configure the Call Bridge through the API rather than the Web Admin Interface. In addition, Call Bridges in a cluster connect to each other over HTTPS, via the Web Admin. From version 2.4, you can improve the security of a Call Bridge cluster by using the Call Bridge trust store to validate Call Bridges within the cluster. See Section 4.12.

Applications that can use internal CA signed certificate	Reason
Call Bridge	The Web Bridge requires and needs to trust a certificate from the Call Bridge. The Active Directory server also needs to trust a certificate from the Call Bridge. In addition, if your deployment has SIP trunks using TLS, then the Call Bridge requires a certificate for mutual authentication with the SIP call control devices.
XMPP server	The Call Bridge validates the identity of the XMPP server.
Load Balancer (only for split server deployments and if native Apps are used)	If native Cisco Meeting Apps are deployed then the trunk between the Core and Edge servers needs to authenticate (trust the certificate) presented by the Load Balancer. Note: the external native Cisco Meeting Apps do not check the Load Balancer certificate.
Trunk (only for split server deployments and if native Apps are used)	If native Cisco Meeting Apps are deployed then the Load Balancer in the Edge server needs to authenticate (trust the certificate) presented by the trunk.
Clustering databases	Database clustering uses public/private key encryption for both confidentiality and authentication. Each server hosting a database requires a set of certificates signed by the same CA. See Section 3.1.5.
Recorder	If you enable a Recorder on the Meeting Server, the Call Bridge requires a signed certificate from the Recorder, and the Recorder requires and needs to trust a certificate from the Call Bridge.
Streamer	If you enable a Streamer on the Meeting Server, the Call Bridge requires a signed certificate from the Streamer, and the Streamer requires and needs to trust a certificate from the Call Bridge.

3 Obtaining certificates

Section explains where certificates are required to establish secure connections in the deployment, and the type of certificates required (signed by a public CA or an internal CA). This chapter focuses on how to obtain the different types of certificate, Chapter 4 covers where to install them.

Note: If you are connecting a Lync deployment to the Meeting Server, you are advised to use the same Certificate Authority (CA) that is trusted by Lync Front End servers. Contact your Lync adviser for details of the CA and for support on the Meeting Server-Lync integration.

All certificates require you following a 3 step process:

1. Generate a private key and the Certificate Signing Request (.csr) file for the specific Meeting Server component.

   Note: The public key is created and held within the .csr file.

2. Submit the .csr file to the CA (public CA or internal CA) for signing.

3. Use SFTP to upload to the Meeting Server the signed certificate and intermediate bundle (if any) from the CA.

The remainder of this chapter provides examples for steps 1 and 2. Chapter 4 covers step 3.

Note: Instructions for generating self-signed certificates using the Meeting Server's MMP commands are provided in Chapter 6. These are useful for testing your configuration in the lab. However, in a production environment you are advised to use certificates signed by a Certificate Authority (CA).

Note: The Meeting Server supports certificates signed using SHA1 and SHA2 algorithms. When the Meeting Server creates certificate signing requests, they are signed using SHA256 in accordance with rules which CAs now operate under.

3.1 Generating a private key and Certificate Signing Request (.csr file)

This section describes using the Meeting Server MMP `pki` command to create a public key and .csr file. If you prefer to use a third party tool to do this, follow the instructions from the third party then resume following this guide from Section 3.2. If you prefer to use OpenSSL to create a private key and .csr file then Appendix A provides an overview of the steps to follow.

You can use the **pki csr <key/cert basename>** command to generate two files: the private key <basename>.key and the certificate signing request file <basename>.csr. They can be immediately retrieved from the Meeting Server by using SFTP.

Note: The basename must NOT include a "." or "_", for example **pki csr basename** is valid, but **pki csr base.name** or **pki csr base_name** are not allowed.

To generate the private key and Certificate Signing Request file:

1. Log in to the MMP
2. Type the **pki csr** command using this syntax
   pki csr <key/cert basename> <CN:value> [OU:<value>] [O:<value>] [ST:<-value>] [C:<value>] [subjectAltName:<value>]

   where

   <key/cert basename> is a string identifying the new key and CSR. Can contain alphanumeric, hyphen or underscore characters.

   CN,OU,O,ST,C,subjectAltName are described in Table 4. Those marked optional can be omitted if you are creating a certificate request file using the pki csr command for signing by a local CA. If you are creating a certificate request file for a public Certificate Authority to sign, you are advised to provide all of the attributes.

Table 4: Attributes in a .csr file

Attribute		Description	Optional/Required
CN	Common Name	This is the fully qualified domain name (FQDN) that specifies the server's exact location in the Domain Name System (DNS). For example, a component with hostname webBridge1 and parent domain example.com has the fully qualified domain name webBridge1.example.com. The FQDN uniquely distinguishes the component from any other components called webBridge1 in other domains.	Required, see notes below
O	Organization or Business name	Usually the legal incorporated name of a company. It should include any suffixes such as Ltd., Inc., or Corp. Use "" around the attribute if more than one word, e.g. "Example Inc."	Optional
OU	Organizational unit or Department name	For example, Support, IT, Engineering, Finance. Use "" around the attribute if more than one word, e.g. "Human Resources"	Optional
L	Location	City or town. For example, London, Boston, Milan, Berlin.	>Optional

Attribute		Description	Optional/Required
ST	Province, Region, County or State	For example, Buckinghamshire, California. Do not abbreviate. Use “” around the attribute if more than one word, e.g. “New Jersey”	Optional
C	Country	The two-letter ISO code for the country where your organization is located. For example, US, GB, FR.	Optional
An email address		An email address to contact the organization. Usually the email address of the certificate administrator or IT department.	Optional
SAN	Subject Alternative Name	From X509 Version 3 (RFC 2459), SSL certificates are allowed to specify multiple names that the certificate should match. This optional field enables the generated certificate to cover multiple domains. It can contain IP addresses, domain names, email addresses, regular DNS host names, etc, separated by commas. If you specify this list you must also include the **CN** in this list. Although this is an optional field, the SAN field must be completed in order for XMPP clients to accept a certificate, otherwise the XMPP clients will display a certificate error. If the Meeting Server is configured to use DNS SRV to lookup the record for the XMPP server, then the SAN field must include the domain name of the XMPP server.	Required for XMPP server certificates or if a single certificate is to be used across multiple components. See note below

Points to note:

- If you plan to use a dedicated certificate for the Web Bridge then specify in the CN field, the FQDN that is defined in the DNS A record for the Web Bridge. Failure to specify the FQDN may result in browser certificate errors.
- If you plan to use a dedicated certificate for the XMPP Server then specify in the CN field, the FQDN that is defined in the DNS SRV record for the XMPP Server. In the **subjectAltName** field specify the domain name of the XMPP server and the DNS SRV record for the XMPP Server.
- If you plan to use the same certificate across multiple components, for example the Web Bridge , XMPP Server, Call Bridge, TURN server, Web Admin, Reorder and Streamer then specify your domain name (DN) in the CN field, and in the SAN field specify your domain name (DN) and the FQDN for each of the components that will use the certificate.
- In the SAN field, ensure there are no spaces between the "," delimiter and the items in the list.

For example:

CN=**example.com**

SAN=
callbridge1.example.com,callbridge2.example.com,callbridge3.example.com,
xmppserver.example.com,webbridge.example.com,example.com

If using the **pki csr** command:

```
pki csr <key/cert basename> <CN:value> [OU:<value>] [O:<value>] [ST:<value>]
[C:<value>] [<subjectAltName:value>]
```

the command is:

```
pki csr onecert CN:example.com
subjectAltName:callbridge1.example.com,callbridge2.example.com,callbridge3.ex
ample.com,xmppserver.example.com,webbridge.example.com
```

Note: If you use the `pki` command, the CN is automatically appended to the SAN list, do not list the CN in the SAN list, as shown in the example above.

3.1.1 CSR for the Call Bridge

Depending on how each Call Bridge is used within your deployment, it may need private key/ certificate pairs:

- to establish communication with the Web Bridge. It is important for the security of the deployment that configuration is only accepted from Call Bridges that are trusted.
- to establish TLS connections with SIP Call Control devices.
- to establish TLS connection with the Lync Front End (FE) server.To ensure that a certificate will be trusted by the Lync FE server:
  - the **CN** in the Certificate must be the same as the FQDN that was added when configuring the Meeting Server as a trusted application on the Lync FE server. To see this value, you can use the Lync Powershell command **`get-cstrustedapplicationcomputer`**.
  - if the certificate has a **`subjectAltName`** list then the FQDN must also be added to the list.
  - sign the certificate using a trusted CA server, such as the CA that has issued the certificates for the Lync FE server.

For example:

```
pki csr callbridge CN:www.example.com O:"Example Inc."
```

or

```
pki csr callbridge CN:www.example.com O:"Example Inc."
subjectAltName:callbridge.example.com
```

The example will generate two files: callbridge.key and callbridge.csr. The files can be immediately retrieved from the Meeting Server by using SFTP. Submit the .csr file to a public CA for signing, see Section 3.2.

Section 4.7 provides details on uploading certificates for the Call Bridges.

3.1.2 CSR for the Web Bridge

Web browsers look at the **CN** field to determine the Web Bridge's FQDN. To avoid web browser certificate errors, follow this advice:

- if you are using a dedicated certificate for the Web Bridge: in the **CN** field, specify the FQDN that is defined in the DNS A record for the Web Bridge. Failure to specify the FQDN may result in browser certificate errors. If the **subjectAltName** field is used, then the FQDN that is specified in the **CN** field needs to be included in the **subjectAltName** field if it is not automatically appended. Note: **pki csr** will automatically append the **CN** to the **SAN** list if a **SAN** list exists.

Note: If your deployment requires the Cisco Expressway Web Proxy to connect to the Web Bridge, then ensure the SAN field of the Web Bridge certificate includes the A record used by the Expressway-C that will connect to the Web Bridge, otherwise the connection will fail. For example, if the Expressway is configured to connect to the Web Bridge on join.example.com, an A record must exist for this FQDN, and the SAN field of the Web Bridge certificate must include join.example.com.

- if you plan to use the same certificate across multiple components (Web Bridge, XMPP Server, Call Bridge and TURN server): specify your domain name (DN) in the **CN** field, and in the **SAN** field specify your domain name (DN) and the FQDN for each of the components that will use the certificate.

For example:

```
pki csr webbridge CN:www.example.com O:"Example Inc."
```

or

```
pki csr webbridge CN:www.example.com O:"Example Inc."
subjectAltName:guest.example.com
```

The example will generate two files: webbridge.key and webbridge.csr. The files can be immediately retrieved from the Meeting Server by using SFTP. Submit the .csr file to a public CA for signing, see Section 3.2.

Section 4.6 provides details on uploading certificates for the Web Bridges.

3.1.3 CSR for the XMPP server

Native Cisco Meeting App looks at the **subjectAltName** and CN fields to determine the XMPP server's domain. To avoid client certificate errors, ensure that the CSR for the XMPP server specifies:

- the DNS record for the XMPP server in the **CN** field, or in **subjectAltName** field
- the XMPP server's domain name in the **subjectAltName** field

For example, if the XMPP domain is configured as example.com and DNS is xmpp.example.com, the CN should be xmpp.example.com and within the SAN list you must add xmpp.example.com and example.com. Note: **pki csr** will automatically append the **CN** to the **SAN** list if a **SAN** list exists.

```
pki csr xmppserver CN:xmpp.example.com O:"Example Inc."
subjectAltName:example.com
```

or using the wildcard:

```
pki csr xmppserver CN:*.example.com O:"Example Inc."
subjectAltName:example.com
```

generates two files: xmppserver.key and xmppserver.csr, for the XMPP server in domain example.com. Submit the .csr file to a public CA for signing, see Section 3.2. The files can be immediately retrieved from the Meeting Server by using SFTP.

Section 4.4 provides details on uploading certificates for the XMPP servers.

Note: When applying certificates for XMPP multi-domains the same requirements apply; the CSR must include both the DNS record for the XMPP server and the domain names for the XMPP multi-domain.

3.1.4 CSR for trunk/Load Balancer pair

In a scalable deployment, with multiple Edge servers, the Core server hosting the XMPP service will require a trunk to each Load Balancer on an Edge server. You need to create a private key/certificate pair for each trunk and Load Balancer. The trunk and the Load Balancer cannot use the same private key/certificate pair, but their certificates can be signed by an internal CA.

1. Create a private key/certificate pair for each Load Balancer enabled on an Edge server.

   For example:
   ```
   pki csr edge1 CN:www.example.com
   ```
   will create edge1.key and edge1.csr.

2. Create a private key/certificate pair for each Core server with the XMPP service enabled.

   For example:
   ```
   pki csr core1 CN:www.example.com
   ```
   will create core1.key and core1.csr.

3. Use an Internal CA to sign edge1.csr and core1.csr certificate signing request files and obtain the corresponding edge1.crt and core1.crt certificates, as well as the Internal CA certificate (bundle). See Section 3.3.

Section 4.5 provides details on uploading certificates for the trunk and Load Balancer.

3.1.5 CSR for database clustering

From version 2.7, database clusters require client and server certificates signed by the same CA configured in each Meeting Server holding or connecting to a database in the cluster. Enforcing the use of certificates ensures both confidentiality and authentication across the cluster.

From 2.7 you can generate all certificates directly from the MMP. Certificates and keys then need to be downloaded via SFTP and uploaded to each Meeting Server that belongs to the same database cluster.

You can still use other methods to create certificates, for example openssl. See Appendix A for more information.

For the database clustering:

1. Create the dbca selfsigned certificate with the pki selfsigned command.

   For example:

   ```
   pki selfsigned dbca CN:"My company CA"
   ```

   creates a local private key named **dbca.key** and a self-signed certificate with the common name **CN=My company CA** in dbca.crt

2. Create a private key and Certificate Request File for the database server. You can use the same certificate on all of the servers in the database cluster; specify the FQDN of one of the servers in the CN field and specify the FQDN of the other servers in the SAN field. If using "Extended Key Usage", ensure "Server Authentication" is allowed for the database server.

   For example:

   ```
   pki csr dbserver CN:server.db.example.com
   subjectAltName:server02.db.example.com
   ```

   generates a CSR file named dbserver.csr and private key named dbserver.key.

3. Create a private key and Certificate Request File for the database client. The CommonName (CN) for a database client must equal 'postgres'. If using "Extended Key Usage", ensure "Client Authentication" is allowed for the database client.

   For example:

   ```
   pki csr dbclient CN:postgres
   ```

   generates a CSR file named dbclient.csr and private key named dbclient.key

4. Use an Internal CA to sign the dbserver.csr and dbclient.csr certificate signing request files and obtain the corresponding dbserver.crt and dbclient.crt certificates, as well as the Internal CA certificate (bundle).

   For example:

   ```
   pki sign dbserver dbca
   pki sign dbclient dbca
   ```

Section 4.8 provides details on uploading certificates for database clustering.

3.1.6 CSR for the TURN server

If you plan to use TLS on a TURN server, then the TURN server will require a certificate/key pair similar to that created for the Web Bridge, so that WebRTC clients trusts the connection. The

certificate should be signed by the same Certificate Authority as used for the Web Bridge certificate.

For example:

```
pki csr turnserver CN:www.example.com O:"Example Inc."
```

The example will generate two files: turn.key and turn.csr. The files can be immediately retrieved from the Meeting Server by using SFTP. Submit the .csr file to a public CA for signing, see Section 3.2.

Section 4.9 provides details on uploading certificates for the TURN servers.

3.2 Signing the CSR using a public Certificate Authority

Refer to Section 1.2 for a list of public CA signed certificates required for the Meeting Server.

To obtain a public CA signed certificate, send the generated .csr file to your preferred Certificate Authority, for example Verisign. The CA will verify your identity and issue a signed certificate. The certificate file will have a .pem, .cer or .crt extension. Appendix B provides a brief overview of file extensions used for certificate files.

Before transferring the signed certificate and the private key to the Meeting Server, check the certificate file. If the CA has issued you a chain of certificates, you will need to extract the certificate from the chain. Open the certificate file and copy the specific certificate text including the BEGIN CERTIFICATE and END CERTIFICATE lines and paste into a text file. Save the file as your certificate with a .crt, .cer or .pem extension. Copy and paste the remaining certificate chain into a separate file, naming it clearly so you recognize it as an intermediate certificate chain and using the same extension (.crt, .cer or .pem). The intermediate certificate chain needs to be in sequence, with the certificate of the CA that issued the chain first, and the certificate of the root CA as the last in the chain.

Go to Chapter 4 for information on installing signed certificates and private keys on the Meeting Server.

Note: In deployments where the Meeting Server is trunked to Cisco Unified Communications Manager, Cisco Unified Communications Manager requires the Call Bridge certificate to be signed using a template that allows for an Extended Key Usage containing both TLS Web Client Authentication and TLS Web Server Authentication.

3.3 Signing the CSR using an internal Certificate Authority

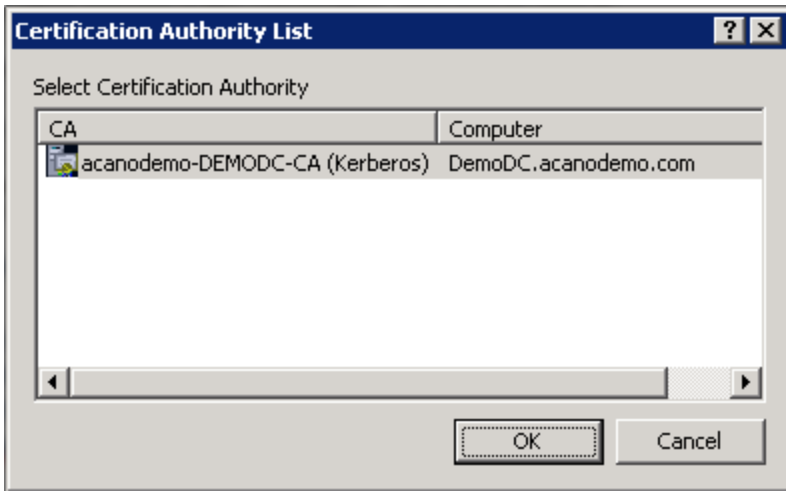
Refer to section Section 1.2 for a list of internal CA signed certificates required for the Meeting Server.

Note: In deployments where the Meeting Server is trunked to Cisco Unified Communications Manager, Cisco Unified Communications Manager requires the Call Bridge certificate to be signed using a template that allows for an Extended Key Usage containing both TLS Web Client Authentication and TLS Web Server Authentication. Microsoft Active Directory Certificate Services can issue this type of certificate.

This section applies if you are using Microsoft Active Directory as an internal CA. If you are using a different internal CA, please follow the corresponding instructions, and then resume following this guide from Chapter 4.

To obtain an internal CA signed certificate, follow these steps:

1. Transfer the generated .csr file to the CA, for example an Active Directory server with the Active Directory Certificate Services Role installed.
2. Issue the following command in the command line management shell on the CA server replacing the path and CSR filename with your information:
   ```
   certreq -submit -attrib "CertificateTemplate:WebServer" <path\csrfilename>
   ```
   For example:
   ```
   certreq -submit -attrib "CertificateTemplate:WebServer"
   C:\Users\Administrator\Desktop\example.csr
   ```
3. After entering the command, a CA selection list is displayed similar to that below. Select the correct CA and click OK.



   If your Windows account has permission to issue certificates, you will be prompted to save the resulting certificate. Save the file with a .crt, .cer or .pem extension, for example example.crt. Go to step 4. See Appendix B for a brief overview of certificate file extensions.

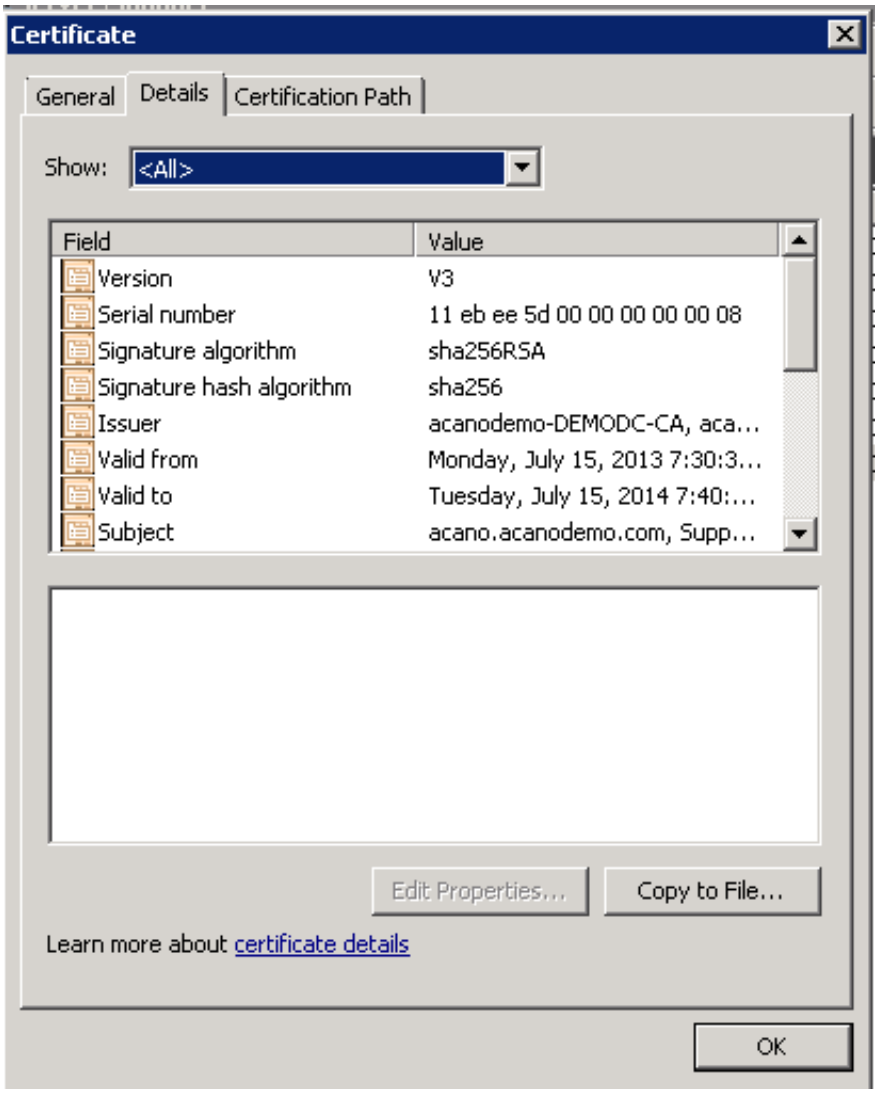
   If you do not see a prompt to issue the resulting certificate, but instead see a message in the command prompt window that the '**Certificate request is pending: taken under submission**', and listing the Request ID, then make a note of the RequestID.

Follow these steps to obtain the issued certificate.

a. Using the **Server Manager** page on the CA, locate the **Pending Requests** folder under the CA Role.

b. Right-click on the pending request that matches the Request ID given in the `cmd` window and select **All Tasks > Issue**.

c. The resulting signed certificate will be in the Issued Certificates folder. Double-click on the certificate to open it and open the **Details** tab.



d. Click **Copy to File** which will start the Certificate Export Wizard.

e. Select Base-64 encoded X.509 (.CER) and click **Next**.

f. Browse to the location in which to save the certificate, enter a name for example **`callbridge`** and click **Next**.

g. Save the resulting certificate with a .crt, .cer or .pem extension, for example **`callbridge.crt`**

4. Go to Chapter 4 for information on installing signed certificates and private keys on the Meeting Server.

4 Installing signed certificates and private keys on the Meeting Server

The scalable and resilient Meeting Server deployment requires public CA signed certificates for:

- the **Web bridge 2**, if Web RTC clients are to be enabled for use by end users. The Web RTC client requires a public CA signed certificate from the Web Bridge in order to trust the connection.
- the **TURN server**, if you plan to use TLS connections for secure communication.
- the **XMPP server**, if native Cisco Meeting Apps (PC, Mac, iOS) are to be used by end users. The Native Cisco Meeting Apps require a public CA signed certificate from the XMPP server in order to trust the connection. This public CA signed certificate can also be used by the Call Bridge and the Web Bridge 2 to validate the identity of the XMPP server, see Section 4.11.

  Note: Web Bridge 3 does not use the XMPP server. For more information, see Appendix D.

- the **Call Bridge**, if direct Lync federation over a public network is required. The Lync Edge server requires a public CA signed certificate from the Call Bridge in order to trust the connection.

And internal CA signed certificates for:

- the **Web Admin**. The Meeting Server API is routed through the Web Admin Interface, so a certificate is required even if you configure the Call Bridge through the API rather than the Web Admin Interface.

  Note: this guide assumes that you have already installed the private key/certificate pair for the Web Admin Interface as described in the Meeting Server Installation Guide. If you have not, do so now.

- the **Call Bridge**. The Web Bridge requires a certificate from the Call Bridge. The Active Directory Server also requires a certificate from the Call Bridge. In addition, if your deployment has SIP trunks, then the Call Bridge requires a certificate for mutual authentication with the SIP call control devices.
- the **Load Balancer**, in split server deployments and if native Cisco Meeting Apps (PC, Mac, iOS) are to be used by end users. The trunk between the Core and Edge server needs to authenticate (trust) the certificate presented by the Load Balancer. This is part of the mutual authentication that needs to be set up between the Load Balancer and the trunk so that the XMPP server trusts the connection. The Load Balancer and the trunk cannot use the same

certificate. Note: the external native Cisco Meeting Apps do not check the Load Balancer certificate.

- the **Trunk**, in split server deployments and if native Cisco Meeting Apps (PC, Mac, iOS) are to be used by end users. The Load Balancer in the Edge server needs to authenticate (trust) the certificate presented by the trunk. This is the other part of the mutual authentication that needs to be set up between the Load Balancer and the trunk. The trunk and the Load Balancer cannot use the same certificate.

- **Database** clustering where each database server and database client (including Call Bridges not co-located with a database) requires a private key and certificate signed by the same Certificate Authority.
- If you enable a **Recorder** on the Meeting Server, the Call Bridge requires a certificate from the Recorder, and the Recorder requires and needs to trust a certificate from the Call Bridge. Refer to the Recorder section in the Cisco Meeting Server Scalable and Resilient deployment guide for details on uploading the certificates and configuring the Recorder.
- If you enable a **Streamer** on the Meeting Server, the Call Bridge requires a certificate from the Streamer, and the Streamer requires and needs to trust a certificate from the Call Bridge. Refer to the Streamer section in the Cisco Meeting Server Scalable and Resilient deployment guide for details on uploading the certificates and configuring the Streamer.

4.1 Reusing a private key and certificate

You do not need to have a different private key/certificate pair for each certificate install. In some circumstances you can copy and reuse the private key and certificate for multiple services. Here is some advice if you reuse a private key/certificate pair:

- if you are connecting a Lync deployment to your Meeting Server, you are advised to use the Certificate Authority (CA) trusted by the Lync deployment.
- use filenames for the certificate and private key that reflect where they are used, for example: **`webadmin.crt`** and **`webadmin.key`**.
- do not use the same private key/certificate pair between a trunk and Load Balancer, they require different private key/certificate pair, see Section 3.1.4.

4.1.1 Example of reusing a private key and certificate

In a deployment with a Load Balancer enabled on two Edge servers and the XMPP service running on a single Core server, you could generate edge1.key, edge1.crt, edge2.key, edge2.crt, core.key, core.crt, and reuse core.key and core.crt on both trunks

4.2 Uploading the Private Keys and Certificates to the MMP

1. SSH into the MMP, and login
2. Use SFTP to upload each private key/certificate pair and certificate bundle
3. Use the MMP PKI command: **`pki list`** to check which files have been uploaded. **`pki list`** will also list any SSH keys and CSR files uploaded to the MMP.

Note: Private keys and certificates must NOT include a "." within the filenames except immediately before the file extension. For example callbridge.key is valid, but call.bridge.key is not allowed.

4.3 Inspecting a file type and checking that a certificate and private key match

Before installing a private key/certificate pair on the Meeting Server, make sure that you have the correct files to install. This section provides a brief overview of using the MMP commands: pki inspect, pki match, and pki verify, to check the identity of the files you plan to install.

To inspect a file to determine whether it is still valid (expiry date):

`pki inspect <filename>`

To check that a certificate matches a private key:

`pki match <keyfile> <certificatefile>`

To check that a certificate is signed by the CA and that the certificate bundle can be used to assert this:

`pki verify <cert> <certbundle/CAcert>`

For example:

1. SSH into the MMP, and login
2. Enter the command:

   `pki inspect callbridge.crt`

   to inspect the contents of the file, for instance to see whether a certificate is still valid.
3. Enter the command:

   `pki match callbridge.key callbridge.crt`

   to check that the file **`callbridge.key`** matches file **`callbridge.crt`** and together they form one usable identity.
4. Enter the command:

   `pki verify callbridge.crt callbridgebundle.crt`

to check that callbridge.crt is signed by a trusted CA, with the chain of trust established through the chain of intermediate certificates in callbridge.crt.

4.4 Installing the Certificate and Private Key for the XMPP Servers

If your Meeting Server supports end users using the native Cisco Meeting Apps for PC, Mac, and iOS devices, then you need to install a public CA signed certificate for the XMPP server. The certificate will be used by the Meeting Apps in the initial setting up of the connection to determine whether they can trust the connection with the XMPP server.

The steps below assume that you have already configured the network interface that the XMPP server will use to listen. Refer to the Scalable and Resilient Server Deployment Guide for information on setting the interface using the **`listen`** MMP command, before assigning the certificate.

1. SSH into the MMP
2. Disable the XMPP server interface before assigning the certificate

   ```
   xmpp disable
   ```
3. Assign the private key/certificate pair using the command:

   ```
   xmpp certs <keyfile> <certificatefile> [<cert-bundle]
   ```

   where **`keyfile`** and **`certificatefile`** are the filenames of the matching private key and certificate . If your CA provides a certificate bundle then also include the bundle as a separate file to the certificate.

   For example:

   ```
   xmpp certs xmppserver.key xmppserver.crt xmppserverbundle.crt
   ```
4. Re-enable the XMPP server interface

   ```
   xmpp enable
   ```

If the certificate installs successfully on the XMPP server, the following is displayed:

```
SUCCESS: Domain configured
SUCCESS: Key and certificate pair match
SUCCESS: license file present
SUCCESS: XMPP server enabled
```

If the certificate fails to install, the following error message is displayed:

```
FAILURE: Key and certificate problem: certificate and key do not match
FAILURE: XMPP server configuration not complete
```

If there is a problem with the certificate bundle, an error message similar to the following is displayed:

```
SUCCESS: Domain configured
SUCCESS: Key and certificate pair match
FAILURE: certificate verification error: depth=x issuer= x = xx, ST = xxxxxxx,
L = xxxxxx, O = xxxxx, OU = xxxxxx, CN = xxxxxxxxx, emailAddress =
xxxxxxxx@xxxxx.xxx Verification error: unable to get issuer certificate Failed
cert:
Certificate:
    Data:
    Version: xx
    Serial Number: xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx
    Signature Algorithm: sha1WithRSAEncryption
    Issuer: C=x, ST= xxxxxxx, L= xxxxxx, O= xxxxx, OU= xxxxxx, CN= xxxxxxxxx,
emailAddress = xxxxxxxx@xxxxx.xxx
    Validity
      Not Before: <month><time><year>
      Not After: <month><time><year>
        Subject: C=xx, O=xxxxx, OU=xxxxxxxxxxx, CN=xxxxxxxxxxxxx
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
SUCCESS: license file present
FAILURE: XMPP server configuration not complete
```

Check the certificate bundle to ensure that there is no break in the certificate chain.

Note: The XMPP license is included in the Cisco Meeting Server software. See the Scalable and Resilient Server Deployment Guide for more information on configuring the XMPP server.

4.5 Installing the Certificate and Private Keys for the Core to Edge Trunks

This section only needs to be followed if your Meeting Server connects to native Cisco Meeting Apps (PC Client, Mac Client or iOS Client) and you have deployed split Core and Edge servers.

Note: In a deployment with split servers, each Core server can create multiple trunks to multiple Edge servers. Each Edge server can accept multiple trunks from multiple Core servers.

To establish mutual authentication between the Load Balancer in the Edge server and the XMPP service in the Core server you need to:

- on the Load Balancer, install the private key/certificate pair for the Load Balancer, and the certificate for the trunk
- on the trunk, install the private key/certificate pair for the trunk, and the certificate for the Load Balancer.

Note: Do not reuse the Load Balancer certificate for the trunk, or vice versa. You must create different certificates for the Load Balancer and for the trunk.

This will ensure that the trunk and the Load Balancer authenticate each other's certificate, creating a secure TLS trunk between the Load Balancer and the XMPP server.

If you have a single XMPP Server in your deployment, but multiple Edge servers, you will quite likely have a Load Balancer on each Edge server. In that case, the Core server (hosting the XMPP service) will need to create a trunk to each of the Load Balancers. Follow steps 1 to 12 below for each trunk/Load Balancer pair. See Section 4.1 for advice on reusing a private key/certificate pair.

Note: This guide assumes that you have already set up the network interface for the Load Balancer and the trunk interface on the Meeting Server, if not follow the steps in the Scalable and Resilient Server Deployment Guide, before assigning the certificate.

---

1. SSH into the MMP of the Edge server
2. Create an Edge instance, using the command:

   `loadbalancer create <tag>`

   If the tag for the Edge server is "Edge1to LB", type:

   `loadbalancer create Edge1toLB`
3. Assign the private key/certificate pair to the Load Balancer and the trunk's certificate using the command:

   `loadbalancer auth <tag> <keyfile> <certificatefile> <trust-bundle>`

   where **`keyfile`** and **`certificatefile`** are the filenames of the matching private key/certificate pair for the Load Balancer, and <trust-bundle> is the certificate for the trunk.

   For example:

   `loadbalancer auth Edge1toLB edge1.key edge1.crt core1.crt`

   Note: The trunk certificate **`core1.crt`** is added as a 'trust bundle' to the Edge server

   ---
4. Configure the trunk interface and port, using:

   `loadbalancer trunk <tag> <iface>:<port>`

   for example, if the trunk connection will be allowed on interface A, port 4999, then type:

   `loadbalancer trunk Edge1toLB a:4999`
5. Configure the public interface and port (for accepting client connections), using

   `loadbalancer public <tag> <iface:port allowed list>`

   for example, if client connections are to be allowed on B, port 5222, then type:
   `loadbalancer public Edge1toLB b:5222`
6. In a common Edge server deployment, the Web Bridge is also enabled and needs to make use of the trunk. To allow this, configure the loopback as a public interface, e.g.
   `loadbalancer public Edge1toLB b:5222 lo:5222`

7. Enable the trunk, using:

   loadbalancer enable <tag>

   for example
   loadbalancer enable Edge1toLB

   Note: The public port is not opened until there is a trunk to service the connection.

8. SSH into the MMP of the Core server
9. Create a trunk between the Core and Edge server for xmpp traffic

   trunk create <tag> <port/service name>

   For example:
   trunk create trunktoEdge1 xmpp
10. Assign the private key/certificate pair to the trunk and the Load Balancer's certificate using the command:

    trunk auth <tag> <key-file> <cert-file> <trust-bundle>

    where **keyfile** and **certificatefile** are the filenames of the matching private key/certificate pair for the trunk, **<trust-bundle>** is the certificate for the Load Balancer.

    For example:

    trunk auth trunktoEdge1 core1.key core1.crt edge1.crt

    Note: The Load Balancer certificate **edge1.crt** is added as a 'trust bundle' to the Core server

11. Configure the Edge server that this trunk will connect to, using:

    trunk edge <tag> <edge name/ip address> [<default port>]

    For example, if the Edge server name is edge1.example.com using port 4999, then type:

    trunk edge trunktoEdge1 edge1.example.com 4999

    Note: If the domain name resolves to multiple IP addresses, a connection will be attempted to all.

12. Enable the trunk interface

    trunk enable <tag>

    For example:

    trunk enable trunktoEdge1

    Note: To see the full list of Load Balancer and trunk commands, see the Cisco Meeting Server MMP Command Reference.

4.6 Installing the Certificate and Private Key for the Web Bridges

If your Meeting Server supports end users using the WebRTC App then you need to install a public CA signed certificate for the Web Bridges. The certificate will be used by browsers when determining whether to trust the connection with the Web Bridge.

The steps below assume that you have already configured the DNS record and the network interface that the Web Bridge will use to listen. Refer to the Scalable and Resilient Server Deployment Guide for information on setting the interface using the **`listen`** MMP command before assigning the certificates.

For each Web Bridge:

1. SSH into the MMP
2. Disable the Web Bridge interface before assigning the certificate
   `webbridge disable`
3. Assign the private key/certificate pairs using the command:

   `webbridge certs <keyfile> <certificatefile> [<cert-bundle>]`

   where **`keyfile`** and **`certificatefile`** are the filenames of the matching private key and certificate. If your CA provides a certificate bundle then also include the bundle as a separate file to the certificate

   For example:
   `webbridge certs webbridge.key webbridge.crt webbridgebundle.crt`
4. Re-enable the Web Bridge interface
   `webbridge enable`

If the certificate installs successfully on the Web Bridge, then the following is displayed:

```
SUCCESS: Key and certificate pair match
SUCCESS: Webbridge enabled
```

If the certificate fails to install, the following error message is displayed:

```
FAILURE: Key and certificate problem: certificate and key do not match
FAILURE: Webbridge configuration not complete
```

Note: Use the MMP command **`webbridge certs none`** to remove the certificate configuration from the Web Bridge.

4.7 Installing the Certificate and Private Key for the Call Bridges

As explained in Section 3.1.1, depending on how each Call Bridge is used within your deployment, it may need private key/ certificate pairs.

The steps below assume that you have already configured the network interface that each Call Bridge will use to listen. Refer to the Scalable and Resilient Server Deployment Guide for

information on setting the interface using the MMP command **listen** before assigning the certificates.

For each Call Bridge:

1. SSH into the MMP of the Meeting Server.
2. Assign the private key/certificate pairs using the command:

   callbridge certs <keyfile> <certificatefile>[<cert-bundle>]

   where **keyfile** and **certificatefile** are the filenames of the matching private key and certificate. If your CA provides a certificate bundle then also include the bundle as a separate file to the certificate.

   For example:

   callbridge certs callbridge.key callbridge.crt callbridgebundle.crt
3. Restart the Call Bridge interface to apply the changes.

   callbridge restart

If the certificate installs successfully on the Call Bridge, then the following is displayed:

```
SUCCESS: listen interface configured
SUCCESS: Key and certificate pair match
```

If the certificate fails to install, the following error message is displayed:

```
FAILURE: Key and certificate problem: certificate and key do not match
```

Note: You will need to add the Call Bridge certificate to every Web Bridge's trust store after you've configured the Web Bridges, see Section 4.7.1.

Note: Use the MMP command **callbridge certs none** to remove the certificate configuration from the Call Bridge.

4.7.1 Establishing Trust between the Call Bridge and the Web Bridge

The Web Bridge allows configuration of guest logins and image customizations to be pushed from a Call Bridge (see the Customization Guidelines document). It is important for the security of the deployment that configuration is only accepted from Call Bridges that are trusted.

Trust between the Call Bridge and the Web Bridge is established by providing the Web Bridge with the public certificate of the Call Bridge. The Web Bridge can use this to challenge the Call Bridge to prove by cryptographic means that it is the owner of the certificate. If the Call Bridge cannot prove that it is the owner of one of the trusted certificates, the Web Bridge will not accept configuration.

Note: In split deployments the Call Bridge certificate needs to be copied from the Core server to the Edge server before you use the **webbridge trust <callbridgecert>** command on the Edge server. On the Core server running Call Bridge:

```
cms>callbridge
Listening interfaces : a
Key file             : callbridge.key
Certificate file     : callbridge.cer
```

Use SFTP to copy the certificate and key file from the Core server to the Edge server running Web Bridge. Then on the Edge server add the certificate to the Web Bridge trust store.

To add the Call Bridge certificate to the Web Bridge trust store:

1. Check which certificate the Call Bridge is using by issuing the **callbridge** command
2. Disable the Web Bridge
3. Add the Call Bridge certificate to the trust store using the command:

   webbridge trust <callbridgecert|cert-bundle>

   and the method below most appropriate to your deployment.

   a. If you only have one Call Bridge and multiple Web Bridges, sign in to each Web Bridge and add the Call Bridge certificate to each Web Bridge's trust store. For example: cms

   ```
   cms>webbridge disable
   cms>webbridge trust callbridge.crt
   cms>webbridge enable
   SUCCESS: Key and certificate pair match
   SUCCESS: webbridge enabled
   ```

   b. If you have several Call Bridges with the same Call Bridge certificate on each Call Bridge then add this Call Bridge certificate to each Web Bridge's trust store. For example:

   ```
   cms>webbridge disable
   cms>webbridge trust callbridge.crt
   cms>webbridge enable
   SUCCESS: Key and certificate pair match
   SUCCESS: webbridge enabled
   ```

   c. If you have several Call Bridges with different certificate files:

      i. Combine all of their certificates into one bundle of trusted certificates, in one of the following ways:

      - Linux or UNIX-like Operating Systems:

        ```
        cat cert1.crt cert2.crt cert3.crt > combinedcallbridgecerts.crt
        ```

      - Windows or DOS:

        ```
        copy cert1.crt + cert2.crt + cert3.crt combinedcallbridgecert.crt
        ```

- Manually combine the certificates using Notepad or Notepad++. There must be no spaces on the first certificate's "END CERTIFICATE" line and the second (and further certificate's) "BEGIN CERTIFICATE" line, but there MUST be a carriage return at the end of the file. They MUST also be in Base64 encoded format.

ii. Then deploy that certificate bundle on each Web Bridge with the command:

```
webbridge trust combinedcallbridgecert.crt
```

4. Re-enable the Web Bridge
5. To verify that the Web Bridge has the Call Bridge certificate in its trust store:

```
cms>webbridge
Enabled                  : true
Interface allowed list     : a:443
Key file                 : webbridge.key
Certificate file         : webbridge.crt
Trust bundle             : callbridge.crt
HTTP redirect            : Enabled
```

4.8 Installing the Certificates and Private Keys for database clustering

CAUTION: The database nodes forming the cluster must be configured with a trusted root CA certificate so that only legitimate nodes can connect to the cluster. The nodes will trust connections that present a certificate chain that ends with a trusted root certificate. Therefore each database cluster must use a dedicated root certificate, the root certificate or intermediate certificates must not be used for any other purpose.

CAUTION: If a database cluster was configured without certificates using a version of Meeting Server earlier than 2.7 (that did not require certificates), then on upgrading to version 2.7 the database will stop and remain unreachable until certificates are configured and the database cluster is recreated.

CAUTION: These pki instructions can be run anytime. However, the database cluster must be disabled for the configuration of the generated certificates (i.e. database cluster certificates). If you have already set up a database cluster you must run the **`database cluster remove`** command on every server in the cluster to disable it, then run the commands to configure the generated certificates before re-creating the cluster.

The certificate for the database client must have CN set to "postgres". You can check that certificate is suitable using the **`pki inspect`** command. For example:

```
cms> pki inspect dbclient.crt
Checking ssh public keys...not found
Checking user configured certificates and keys...found
File contains a PEM encoded certificate
```

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            58:00:00:00:1c:3b:92:8a:95:d2:21:89:58:00:00:00:00:00:1c
    Signature Algorithm: sha1WithRSAEncryption
        Issuer: DC=com, DC=support, CN=support-DC2-CA
        Validity
            Not Before: Sep 13 13:32:38 2015 GMT
            Not After : Sep 13 13:42:38 2017 GMT
        Subject: CN=postgres
```

If you have CN set to anything other than postgres for the database client certificate you will see the error message: "ERROR: Client certificate common name incorrect" when configuring the cluster.

Note: the **`database cluster initialize`**, **`database cluster join`** and **`database cluster connect`** commands will not run without valid certificates, keys and CA certificates uploaded to the database clients and servers. The error message: **`FAILURE: certificates must be configured`** displays to indicate that certificates have not been configured on this database server or client.

4.8.1 Uploading certificates for a database cluster

You now need to upload the certificates to the Meeting Server and then configure clustering.

1. On the host server of every database (whether co-located with a Call Bridge or not):
    a. SFTP the following certificate and keys to the Call Bridge server:
        - dbserver.key
        - dbserver.crt
        - dbclient.key
        - dbclient.crt
        - the certificate bundle provided by the internal CA. As per this example, the file will be dbca.crt
    b. Specify the certificates that will be used when the database cluster is built. For example:

```
cms>database cluster certs dbserver.key dbserver.crt dbclient.key
dbclient.crt dbca.crt
Certificates updated
cms> database cluster status
Status                    : Disabled
Interface                 : a
Certificates
  Server Key              : dbserver.key
```

```
Server Certificate      : dbserver.crt
Client Key              : dbclient.key
Client Certificate      : dbclient.crt
CA Certificate          : dbca.crt
```

Note: The **database cluster status** command will highlight if there is a lack of configured certificates.

2. For every Call Bridge not co-located with a database:
    a. SFTP the following certificate and keys to the Call Bridge server:
        - dbclient.key
        - dbclient.crt
        - the certificate bundle provided by the internal CA. As per this example, the file will be dbca.crt
    b. Configure the database cluster to use these certificates:

```
cms> database cluster certs dbclient.key dbclient.crt dbca.crt
Certificates updated
cms> database cluster status
Status                  : Disabled
Interface               : a
Certificates
  Client Key            : dbclient.key
  Client Certificate    : dbclient.crt
  CA Certificate        : dbca.crt
```

3. Now return to the Scalable and Resilient Server Deployment Guide for information on selecting the master database and building the database cluster.

4.9 Installing the Certificates and Private Keys for TURN Servers

If you plan to use TLS for secure communication, then you need to install a signed certificate for the TURN servers, using the same CA to sign the certificate as that used for the Web Bridge. The certificate will be used by browsers when determining whether to trust the connection with the Meeting Server.

The steps below assume that you have already configured the network interface that the TURN server will use to listen. Refer to the Scalable and Resilient Server Deployment Guide for information on setting the interface using the **listen** MMP command before assigning the certificates.

For each TURN server:

1. SSH into the MMP of the host server
2. Disable the TURN server interface before assigning the certificate

```
turn disable
```

3. Use SFTP to upload to the Meeting Server the signed certificate and intermediate bundle (if any) from the CA.
4. Check that the certificate (and certificate bundle) and the private key match

   ```
   pki verify <certicate> <cert bundle/CA cert> [<CA cert>]
   ```

5. Assign the certificate (and certificate bundle) and private key pair to the Turn server

   ```
   turn certs <keyfile> <certificatefile> [<cert-bundle>]
   ```

   where keyfile and certificatefile are the filenames of the matching private key and certificate. If your CA provides a certificate bundle then also include the bundle as a separate file to the certificate

   For example

   ```
   turn certs turn.key turn.crt turnbundle.crt
   ```

6. Re-enable the TURN server

   ```
   turn enable
   ```

4.10 TLS Certificate Verification

You can enable Mutual Authentication for SIP and LDAP in order to validate that the remote certificate is trusted. When enabled, the Call Bridge will always ask for the remote certificate (irrespective of which side initiated the connection) and compare the presented certificate to a trust store that has been uploaded and defined on the server.

The MMP commands available are:

- to display the current configuration

  ```
  tls <sip|ldap>
  ```

- to define the Certificate Authorities to be trusted

  ```
  tls <sip|ldap> trust <crt bundle>
  ```

- to enable/disable certificate verification or whether OCSP is to be used for verification

  ```
  tls <sip|ldap> verify enable|disable|ocsp
  ```

See the MMP Command Reference guide for further information.

4.11 XMPP server certificate validation

From version 2.4.0, the Call Bridge and Web Bridge have trust stores to hold the certificates for the XMPP servers in the deployment. If configured, these trust stores enable the Call Bridge and Web Bridge to check the identity of the XMPP servers when making connections to them. Validating the certificate files of the XMPP servers ensures that XMPP servers are legitimate, and removes the risk that an attacker could redirect traffic to an insecure XMPP server. In addition,

validation can be used to prevent the WebRTC app from being used to connect to meetings hosted by other Meeting Server deployments.

Note: By default, neither the Web Bridge nor the Call Bridge validate the certificate files of the XMPP servers in a deployment. If you choose to use this feature then we recommend that you configure validation by both the Web Bridge and the Call Bridge, see below.

CAUTION: If you enable XMPP server validation, but have added a certificate bundle to the Web Bridge and Call Bridge trust stores that is not properly configured, then participants using Cisco Meeting App will be unable to join the Meeting Server meetings.

4.11.1 Single XMPP server deployment

If you are using a single XMPP server, .

1. Use the MMP command **`xmpp`** or **`xmpp status`** to identify the name of the XMPP server's certificate file:

   ```
   CMS-SERVER> xmpp
   Enabled : true
   Clustered : false
   Domain : cms.example.com
   Listening interfaces : a
   Key file : private.key
   Certificate file : xmpp-certificate.crt
   Max sessions per user : unlimited
   STATUS : XMPP server running
   ```

   Note: To avoid certificate errors, ensure that the XMPP server certificate specifies:
   - the XMPP server's domain name (FQDN) in the subjectAltName field. DO NOT use an IP address in the subjectAltName field.

   ---

2. Disable the Web Bridge before uploading the certificate bundle.

   `webbridge disable`

3. Use the MMP command **`webbridge trust xmpp <xmpp certificate bundle>`** to upload the XMPP server's certificate file to the Web Bridge trust store.

   `webbridge trust xmpp xmpp-certificate.crt`

4. Re-enable the Web Bridge.

   `webbridge enable`

5. Use the MMP command **`callbridge trust xmpp <xmpp certificate bundle>`** to upload the XMPP server's certificate file to the Call Bridge trust store.

   `callbridge trust xmpp xmpp-certificate.crt`

Note: It is not mandatory to upload the XMPP server certificate file to both the Web Bridge and the Call Bridge, however it is advised for maximum security.

After uploading the certificate file of the XMPP server to the trust stores of the Web Bridge and the Call Bridge, both the Web Bridge and the Call Bridge will validate the certificate received from the XMPP server to ensure that they only establish a connection to the XMPP server whose certificate they hold in their trust store.

4.11.2 Resilient XMPP server deployment

If you have deployed more than one XMPP server for failover and resiliency, then you will have already created a bundle of XMPP server certificates; see section "Example of deploying XMPP resiliency" in the Scalability and Resilient Deployment guide. If you choose to configure XMPP server validation then the certificate bundle needs to be copied onto every Meeting Server running either the Web Bridge or Call Bridge.

1. Use the MMP command **`xmpp cluster status`** to identify the name of the trust bundle holding the certificates for the cluster of XMPP servers:

```
CMS-SERVER> xmpp cluster status
Last state change: 2018-Aug-17 13:42:12
Key file : private.key
Certificate file : xmpp-certificate.crt
Trust bundle : cmstrust.crt
```

Note about split Meeting Server deployments: In a split deployment, the XMPP server is located on the core server with the XMPP Load Balancer on an edge server. The XMPP Load Balancer acts as a gateway for the current XMPP leader at that time. When the Web Bridge connects to the XMPP Load Balancer, the Web Bridge needs to receive the certificate of the XMPP server that resides on the core server, rather than the certificate of the edge server. Consequently, the trust bundle used in **`webbridge trust xmpp <xmpp certificate bundle>`** must contain the certificates of the core XMPP server(s).

```
EDGE-0> webbridge
Enabled                    : true
Interface allowed list     : a:443
Key file                   : private.key
Certificate file           : edge-0cms.crt
CA Bundle file             : CA.crt
XMPP Trust bundle          : cmscluster.crt
HTTP Trust bundle          : cmscluster.crt
HTTP redirect              : Enabled
HTTP URL redirect          : true
.....
```

Note: To avoid certificate errors, ensure each XMPP server certificate specifies:
- the XMPP server's domain name in the subjectAltName field.

2. Use the MMP command **`webbridge trust xmpp <xmpp cluster certificate bundle>`** to upload to the Web Bridge trust store the bundle of certificates for the cluster of XMPP servers.

   `webbridge trust xmpp cmstrust.crt`

3. Use the MMP command **`callbridge trust xmpp <xmpp cluster certificate bundle>`** to upload to the Call Bridge trust store the bundle of certificates for the cluster of XMPP servers.

   `callbridge trust xmpp cmstrust.crt`

4.11.3 Removing certificate validation

To stop the Web Bridge validating the XMPP server's certificate, use the MMP command:

`webbridge trust xmpp none`

To stop the Call Bridge validating the XMPP server's certificate, use the MMP command:

`callbridge trust xmpp none`

Note: Currently, if the trust stores are not used then the Web Bridge and the Call Bridge will continue to make connections to the XMPP server(s) without verifying the server(s) identity.

4.12 Call Bridge cluster validation

You can improve the security of a Call Bridge cluster by using the Call Bridge trust store to validate Call Bridges within the cluster. As Call Bridges connect to each other over HTTPS, which is fronted by the Web Admin, you need to create a certificate bundle holding the Web Admin certificates of the clustered Call Bridges, and upload the certificate bundle to the trust store of each Call Bridge in the cluster. Use the MMP command:

`callbridge trust cluster <bundle name>`

When a Call Bridge connects to another Call Bridge in a cluster, it checks the bundle of certificates in its trust store to validate the identity of the Call Bridge that it is connecting to. This removes the risk that the Call Bridge is connecting to an insecure Meeting Server. The certificate bundle can be either a certificate chain or an allowed list of trusted certificates.

If the trust store is not used, then there will be no certificate validation between clustered Call Bridges, and a Call Bridge will continue to make the connection to a remote Call Bridge, but without verifying its identity.

To remove the Call Bridge cluster certificate bundle from the Call Bridge trust store, use the MMP command:

`callbridge trust cluster none`

5 Troubleshooting problems with certificates

This section covers a few common troubleshooting problems. Refer to the Meeting Server Knowledgebase for further Frequently Asked Questions relating to certificates.

5.1 Warning message that service is untrusted

The message is displayed if:

- you have used an internal CA which is not in your trust store,
- you have used a self-signed certificate where a public or internal CA signed certificate is required. Re-issue the certificate and have it signed by a trusted CA: this can be an internal CA unless you want public access to this component.

5.2 Client certificate error

In order for the clients to communicate with the XMPP server they must trust the connection. For the client to trust the connection, and avoid client certificate errors, the certificate for the XMPP server should include:

- the DNS record for the XMPP server in the **`CN`** field,
- the XMPP domain name in the **`subjectAltName`** field

For example, if clients are using 'firstname.surname@example.com', then the XMPP domain is example.com and DNS is xmpp.example.com. The CN should specify xmpp.example.com and within the SAN list you must add example.com. The CN is automatically appended to the SAN list.

```
pki csr xmppserver CN:xmpp.example.com O:"Example Inc."
subjectAltName:example.com
```

or using the wildcard:

```
pki csr xmppserver CN:*.example.com O:"Example Inc."
subjectAltName:example.com
```

5.3 Browser certificate error

In order for the WebRTC clients to communicate with the Web Bridge 2 they must trust the connection. For the client to trust the connection, the certificate for the Web Bridge 2 must include the Fully Qualified Domain Name (FQDN) that specifies the server's exact location in the Domain Name system (DNS). It will have been defined in the DNS A record for the Web Bridge 2 service. Failure to specify the FQDN in the Web Bridge 2 certificate will result in browser certificate errors.

In addition, if TLS is configured on the Meeting Server, then the TURN servers require a similar certificate to the certificate/key pair assigned to the Web Bridge 2. Failure to assign a suitable certificate/key pair to the TURN servers may result in the client not trusting the connection and the browser displaying a certificate error.

5.4 Call Bridge cannot connect to Web Bridge 2

Either the Web Bridge 2 does not have the certificate of the Call Bridge in its trust store or the certificate has expired (Authentication error).

Type **`webbridge`** to show the certificate that is in the Web Bridge's trust bundle.

Type **`pki inspect <certificate name>`** to show you the validity of the certificate.

5.5 Problem connecting to Lync Front End server

Check that the CA that signed the Call Bridge certificate is the same CA that was used to sign the certificate for the Lync Front End server. If the Call Bridge certificate is not signed by the same CA that was used to sign the certificate for the Lync Front End server, then make sure the Lync server can trust the Call Bridge certificate by uploading the Call Bridge Trusted CA certificate to the Root Trust Store of the Lync Servers.

Ensure that the FQDN that was added on Lync, is also present as the CN in the Call Bridge's certificate.

5.6 Certificate soon to expire or already expired message

To renew a certificate is exactly the same process as how you deploy certificates in the first instance. Refer back to the earlier sections on obtaining and installing certificates.

6 Creating and using certificates in a test environment

You can create a private key and self-signed certificate on the Meeting Server using the **pki selfsigned** command.

Self-signed certificates cannot be used for 'cluster keys' (as no CA certificate can be obtained) or Lync authentication (as CA is not a trusted authority). But self-signed certificates can be used for Web Admin, for trunk/Load Balancer, and mutual authentication between the Call Bridge and Web Bridge 2, although the browser will display a certificate error. It is strongly recommended that you use self-signed certificates in a test environment, rather than in a production environment.

To generate a local private key and a self-signed certificate on the Meeting Server:

1. Log in to the MMP and type the command:

   pki selfsigned <key/cert basename>

   where **<key/cert basename>** identifies the key and certificate which will be generated.

   For example:

   pki selfsigned callbridge

   creates a local private key named **callbridge.key** and a self-signed certificate named **callbridge.crt**

Appendix A OpenSSL Commands for Generating Certificates

Instead of the MMP **pki** command described in Section 1.2, OpenSSL can be used to generate private keys, certificate signing requests and certificates. This appendix details the OpenSSL commands to use. The examples assume OpenSSL is running on Windows, although OpenSSL can be used on other platforms.

Note: Run OpenSSL in Administrator mode.

Note: The examples in this chapter use Web Bridge 2.

A.1 Generating RSA private keys and CSR files

Use the OpenSSL toolkit on your computer.

To generate a new RSA private key and CSR file, use the command:

```
openssl req -out <certname>.csr -new- newkey rsa:2048 -addext
"keyUsage=digitalSignature" -nodes -keyout <keyname>.key
```

For example:

```
openssl req -out webbridge.csr -new -newkey rsa:2048 -addext
"keyUsage=digitalSignature" -nodes -keyout webbridge.key
```

generates a CSR file named webbridge.csr and an RSA 2048 bit private key named webbridge.key,

Note: The keyname and certname must NOT include a "." or "_", for example **webbridge** is valid, but **web.bridge** or **web_bridge** are not allowed.

To generate a certificate signing request (CSR) for an existing private key, use the command:

```
openssl req -out <certname>.csr -key <keyname>.key –new
```

For example:

```
openssl req -out webbridge.csr -key webbridge.key –new
```

generates a CSR file named webbridge.csr based on an existing private key named webbridge.key,

If you intend to self-sign the certificate using OpenSSL, then no intermediate CSR file is required, go to the next section.

A.2 Signing CSR files

To sign the CSR using a public CA follow the instructions in Section 3.2.

To sign the CSR using an internal CA follow the instructions in Section 3.3.

To self-sign the certificate, use the OpenSSL command:

```
openssl req -x509 -nodes -days 100 -newkey rsa:2048 -keyout <keyname>.key -out <certname>.crt
```

For example:

```
openssl req -x509 -nodes -days 100 -newkey rsa:2048 -keyout callbridge.key -out callbridge.crt
```

generates a new private key named callbridge.key and a (final) certificate named callbridge.crt.

A.3 Creating Certificates for Database Clustering

This section details how to create certificates for database clustering using OpenSSL commands.

Note: As database cluster certificates are mandatory from 2.7, to make it easier to setup Meeting Server database clustering you can use pki commands to create signed certificates for the database cluster. For more information, see Section 3.1.5.

The certificates created for database clustering must be signed by the same Certificate Authority (CA). Because the database clustering is not user-accessible, the CA, keys and certificates can be generated internally using OpenSSL.

Database clusters require client and server certificates signed by the same CA configured in each Meeting Server holding or connecting to a database in the cluster. Enforcing the use of certificates ensures both confidentiality and authentication across the cluster.

CAUTION: If a database cluster was configured without certificates using an earlier version of Meeting Server software which did not require certificates, then on upgrading to version 2.7 the database will stop and remain unreachable until certificates are configured and the database cluster is recreated.

Note: Ensure that you run OpenSSL with administrator privileges.

Note: The keyname and certname must NOT include a "." or "_", for example **`db01_ca`** or **`db01.ca`** are not allowed.

Follow these steps:

1. Define a CA, and create the private/public key pair and certificate for the CA.

a. Generate a private key and certificate request (.csr) pair for a CA defined by you. Use this OpenSSL syntax

```
openssl req -new -text -nodes -keyout <keyname>.key -out <certname>.csr
-subj /C=<country>/ST=<state>/L=<location>
/O=<organization>/OU=<organizational unit>/CN=<authorityname>
```

For example:

```
openssl req -new -text -nodes -keyout db01ca.key -out db01ca.csr -subj
/C=UK/ST=London/L=London/O=Example/OU=/CN=example
```

creates the private key **db01ca.key** and the certificate signing request file **db01ca.csr** for the CA defined in the attributes following **-subj**

b. Create a certificate for the CA, using the private key and certificate request (.csr) generated in step 1a.

```
openssl req -x509 -text -in db01ca.csr -key db01ca.key -out db01ca.crt -
days 3650
```

creates the certificate **db01ca.crt**

2. Use the CA credentials generated in step 1 to output a private key and signed certificate for the database server and database client.

a. Generate a private key and certificate request (.csr) for the database server:

```
openssl req -new -nodes -keyout <keyname>.key -out <certname>.csr –subj
/C=<country>/ST=<state>/L=<locality>
/O=<organization>/OU=<organizational unit>/CN=<nodename>
```

where **nodename** is the actual name of the server hosting the database. For example:

```
openssl req -new -nodes -keyout db01server.key -out db01server.csr -subj
/C=UK/ST=London/L=London/O=Example/OU=/CN=server1
```

creates the key **db01server.key** and the certificate signing request file **db01server.csr**

b. Generate the CA signed certificate for the database. For example:

```
openssl x509 -req -CAcreateserial -in db01server.csr -CA db01ca.crt -
CAkey db01ca.key -out db01server.crt -days 3650
```

creates the certificate **db01server.crt**

c. Generate a private key and certificate request (.csr) for the database. The CommonName (CN) for a database client must equal “postgres”.

```
openssl req -new -nodes -keyout <keyname>.key -out <certname>.csr
-subj /C=<country>/ST=<state>/L=<locality>/O=<organization>
/OU=<organizational unit>/CN=postgres
```

For example:

```
openssl req -new -nodes -keyout db01client.key -out db01client.csr -subj
/C=UK/ST=London/L=London/O=Example
```

```
/OU=/CN=postgres
```

creates the key **db01client.key** and the certificate signing request file **db01client.csr**.

d. Generate the CA signed certificate for the database client. For example:

```
openssl x509 -req -CAcreateserial -in db01client.csr -CA db01ca.crt -
CAkey db01ca.key -out db01client.crt -days 3650
```

creates the certificate **db01client.crt**

3. Follow the steps in Section 4.8 to upload and assign the database certificates and private keys .

   a. Each server hosting a database, requires the following keys and certificates to be uploaded:
      - database cluster server certificate (generated in step 2)
      - database cluster server key (generated in step 2)
      - database cluster client certificate (generated in step 2)
      - database cluster client key (generated in step 2)
      - database cluster CA certificate bundle (generated in step 1)

   b. Each Call Bridge NOT co-located with a database, requires the following keys and certificates to be uploaded:
      - database cluster client certificate (generated in step 2)
      - database cluster client key (generated in step 2)
      - database cluster CA certificate bundle (generated in step 1)

A.4 Installing Certificate and Private Key Pairs

For details on installing the certificate and private key pairs on the Meeting Server, follow the instructions in Chapter 4.

Appendix B Permitted extensions for certificate files and private keys

The following tables list the permitted file extensions for certificate files and private keys.

Table 5: Permitted extensions for certificate files

Extension	Information on file type
.pem	PEM is both an encoding (ASCII base64) and used as a file extension. Typically imported from a Unix-based Apache Web server and compatible with OpenSSL applications. PEM certificate files are generated automatically. Some secure websites may ask users to upload a PEM file (possibly sent in an e-mail) in order to authenticate their identity.
.der	Distinguished Encoding Rules (DER) is both an encoding and used as a file extension. Contains a binary representation of the certificate created in the DER format. Commonly used for storing X.509 certificates in public cryptography.
.cer	Security file provided by a third party Certificate Authority, such as VeriSign or Thwate, that verifies the authenticity of a website. Installed on a Web server to establish the validity of a specific website hosted on the server. The certificates may be encoded as binary DER or as ASCII (Base64) PEM.
.crt	Certificate used by secure websites (beginning with "https://") to verify their authenticity. Distributed by companies such as Verisign and Thawte. The certificates may be encoded as binary DER or as ASCII (Base64) PEM. Certificate files are automatically recognized by Web browsers when a user visits a secure site. The information stored in the certificate can be viewed by clicking the lock icon within the browser window.

Table 6: Permitted extensions for private key files

Extension	Information on file type
.key	Used both for public and private PKCS#8 keys. The keys may be encoded as binary DER or as ASCII PEM.
.pem	Indicates key was encoded using PEM (ASCII base64).
.der	Indicates key was encoded using binary DER.

Appendix C MMP PKI commands

Below is a list of MMP pki commands:

Command/Examples	Description/Notes
pki	Displays current PKI usage.
pki list	Lists PKI files i.e. private keys, certificates and certificate signing requests (CSRs).
pki inspect <filename>	Inspect a file and shows whether the file is a private key, a certificate, a CSR or unknown. In the case of certificates, various details are displayed. If the file contains a bundle of certificates, information about each element of the bundle is displayed. Both PEM and DER format files are handled.
pki match <key> <certificate>	This command checks whether the specified key and a certificate on the system match. A private key and a certificate are two halves of one usable identity and must match if they are to be used for a service e.g. callbridge.
pki verify <cert> <cert bundle/CA cert> [<CA cert>] **pki verify server.pem bundle.pem rootca.pem** **pki verify server.pem bundle.pem**	A certificate may signed by a certificate authority (CA) and the CA will provide a "certificate bundle" of intermediate CA certificates and perhaps a CA certificate in its own file. To check that the certificate is signed by the CA and that the certificate bundle can be used to assert this, use this command.
pki unlock <key>	Private keys are often provided with password-protection. To be used in the Meeting Server, the key must be unlocked. This command prompts for a password to unlock the target file. The locked name will be replaced by an unlocked key with the same name

Command/Examples	Description/Notes
pki csr <key/cert basename> [<attribute>:<value>] **pki csr example CN:www.example.com OU:"My Desk" O:"My Office" L:"San Jose" ST:California C:US**	For users happy to trust that Cisco meets requirements for generation of private key material, private keys and associated Certificate Signing Requests can be generated. <key/cert basename> is a string identifying the new key and CSR (e.g. "new" results in "new.key" and "new.csr" files) Attributes for the CSR can be specified in pairs with the attribute name and value separated by a colon (":"). Attributes are: CN: commonName which should be on the certificate. The commonName should be the DNS name for the system. OU: Organizational Unit O: Organization L: Locality ST:State C: Country emailAddress: email address The CSR file can be downloaded by SFTP and given to a certificate authority (CA) to be signed. (Alternatively, the CSR file can be used in the 'pki sign' command to generate a certificate locally.) On return it must be uploaded via SFTP. It can then be used as a certificate. Note: Since 1.6.11 **pki csr <key/cert basename> [<attribute>:<value>]** now takes subjectAltName as an attribute. IP addresses and domain names are supported for subjectAltName in a comma separated list. For example: **pki csr test1 CN:example.exampledemo.com subjectAltName:exampledemo.com** **pki csr test2 CN:example.exampledemo.com C:US L:Purcellville O:Example OU:Support ST:Virginia subjectAltName:exampledemo.com** **pki csr test3 CN:example.exampledemo.com C:US L:Purcellville O:Example OU:Support ST:Virginia subjectAltName:exampledemo.com, 192.168.1.25,xmpp.exampledemo.com, server.exampledemo.com,join.exampledemo.com, test.exampledemo.com** Keep the size of certificates and the number of certificates in the chain to a minimum; otherwise TLS handshake round trip times will become long.

Command/Examples	Description/Notes
pki selfsigned <key/cert basename> [<attribute>:<value>]	You can use this command to generate self-signed certificates. <key/cert basename> identifies the key and certificate which will be generated, e.g. "pki selfsigned new" creates new.key and new.crt (which is self-signed). Attributes for the CSR can be specified in pairs with the attribute name and value separated by a colon (":"). Attributes are: CN: commonName which should be on the certificate. The commonName should be the DNS name for the system. OU: Organizational Unit O: Organization L: Locality ST:State C: Country emailAddress: email address The CSR file can be downloaded by SFTP and given to a certificate authority (CA) to be signed. On return it must be uploaded via SFTP. It can then be used as a certificate. Keep the size of certificates and the number of certificates in the chain to a minimum; otherwise TLS handshake round trip times will become long.
pki sign <csr/cert basename> <CA key/cert basename>	This command signs the csr identified by <csr/cert basename> and generates a certificate with the same basename, signed with the CA certificate and key identified by <CA key/cert basename>. The files <csr/cert basename> and <CA key/cert basename> should have been generated by the commands 'pki csr' and 'pki selfsigned' respectively.
pki pkcs12-to-ssh <username> **pki pkcs12-to-ssh john**	Public SSH keys stored in PKCS#12 files can be used but need to be processed first. This command extracts a useable public key from a PKCS#12 file uploaded with the name <username>.pub.You are prompted to enter the password for the pkcs#12 file. After completion, the pkcs#12 file is replaced with a useable key without password protection. Note: Any other data contained in the pkcs#12 file is lost. The key of an uploaded PKCS#12 file john.pub for user john can be made useable by executing this command

Appendix D Certificate and configuration information to deploy Web Bridge 3 to use Cisco Meeting Server web app

In version 2.9, Meeting Server introduces the new Cisco Meeting Server web app which is a browser-based client for Cisco Meeting Server that lets users join meetings (audio and video). To use this feature you need to deploy the new Web Bridge 3. In addition, Meeting Server version 2.9 still offers the original Cisco Meeting App WebRTC (also referred to here as Web Bridge 2).

In this release Cisco Meeting Server web app is fully supported for internal calls, but not recommended for external calls (see the 2.9 Release Notes for more information), and it is not yet fully featured. It is intended that in due course it will support virtually the same feature set and supersede Cisco Meeting App WebRTC.

Note: For a full list of features that are not currently supported by web app in 2.9 and those features that we plan to support in the future, see Cisco Meeting Server web app Important Information for more details.

Note: The Web Bridge 3 component that supports web app cannot be run on the Acano X-series. However, the Acano X-Series can still run the Call Bridge and be part of the same cluster. Web Bridge 3 will need to be run on Cisco Meeting Server 1000 and 2000 platforms and other specification-based VM.

Note: Web app does not require XMPP. The XMPP component will be removed from a future version. Cisco Meeting App WebRTC still requires XMPP.

CAUTION: Important notes for Expressway users
If you are deploying Web Bridge 3 and web app you must use Expressway version X12.6 or later, earlier Expressway versions are not supported by Web Bridge 3. If you are deploying solely Web Bridge 2 and Meeting App for WebRTC you can continue to use Expressway versions earlier than X12.6.

D.1 Useful information to help configure Web Bridge 3

The following is useful information to help you configure Web Bridge 3 so that you can use web app:

- Web Bridge 3 requires all of its certificate definitions to use a bundle of certificates, i.e. a full chain file. This is different to how certificates are implemented for Web Bridge 2.
- "Call Bridge to Web Bridge" protocol (C2W) is the link between the callbridge and webbridge3.
- A port must be opened on an interface (using **`webbridge3 c2w listen`**) to allow the callbridge to connect to the webbridge3 (the webbridge listens on that port). This is why you have to give the address with this port when you do the API request to tell a callbridge about this webbridge. This connection must be secured with certificates.
- We recommend you protect that opened port from external access – it only needs to be reachable from callbridges.
- The callbridge uses the certificate set using **`callbridge certs`** and the webbridge uses the certificate set using **`webbridge3 c2w certs`**.
- The webbridge will trust certificates of callbridges that have been signed by one of those in its trust store, set by **`webbridge3 c2w trust`**.
- The callbridge will trust webbridges that have certificates signed by one of those in its trust store, set by **`callbridge trust c2w`**.
- The webbridge3 https certifcates and ports are the same as for webbridge2, it allows you to reach the web client using https and can be used in the same deployment at the same time.
- If the webbridge3 c2w certificate requires extended key usage, it should be "server authentication", and the callbridge certificate extended key usage should be "client authentication". However, these extensions are optional and if the certificate doesn't have them, the Web Bridge 3 will assume any usage is possible.
- You do not need a certificate signed by a public authority – you can use self-signed certificates created within the MMP.
- The SAN/CN must match the FQDN or IP address that is used in the c2w:// url used to register the Web Bridge 3 in the callbridge API. (If this does not match, the callbridge will fail the TLS negotiation, rejecting the certificate presented by the webbridge, and will fail to connect with the webbridge.)

D.2 Generating certificates for Meeting Server

Meeting Server uses x.509 certificates to configure secure (TLS) connections for its services and for some authentication tasks. In this deployment, certificate configuration is required for the Call Bridge, Web Bridge, and Web Admin services. Certificates can be self-signed or signed by internal or external certificate authorities.

For this simplified deployment we will use one x.509 certificate with the correct attributes signed by an internal or external Certification Authority (CA). Using a self-signed certificate here

is possible but is not recommended as it will cause errors to be seen in web pages and will prevent you from incorporating Meeting Server into Unified CM as a conference bridge.

For this deployment, our certificate should have the server FQDN as the Common Name (CN) and must be in the Subject Alternate Name (SAN) attribute of the certificate. The "digitalSignature" key usage bit must also be set. To generate a Certificate Signing Request (CSR) and private key using the MMP:

1. Log in to the MMP using SSH or console.
2. Enter the **pki csr** command using this syntax:
   pki csr <key/cert basename> <CN:value> [OU:<value>] [O:<value>] [ST:<-value>] [C:<value>] [subjectAltName:<value>]

   For example:

   pki csr singleCert CN:meetingserver.company.com

   Note: The **CN,OU,O,ST,C** values and other attributes are optional in the certificate and are omitted here for simplicity. They can be defined and included if desired.

   Note: The CN value should always be part of the SubjectAltName (SAN) list. The Meeting Server **pki csr** command adds the CN to the SAN list automatically so you do not have to list it separately.

   The output of this command generates a private key file with the extension **.key** and a Certificate Signing Request (CSR) file with the extension **.csr** on the Meeting Server's file system.
3. Using your SFTP client, log into Meeting Server and copy the CSR file to your machine so it can be supplied to your signing certificate authority.
4. Supply the CSR file to your certificate authority to be signed. They will return a signed certificate in a binary or text encoded (PEM) format (for example, singleCert.crt). This guide will use examples using PEM files. Other formats are supported but not documented here. Obtaining your files in PEM format or converting them to PEM is recommended for simplicity.
5. To complete the installation, you will need three variations of your certificate files. You will need the signed certificate itself (example: singleCert.crt), the Root & Intermediate certificate bundle from your CA (example: ca-bundle.crt), and a full chain file version of your certificate (example: single-chain.crt).

   The Root & Intermediate certificate bundle includes the chain of trust that signed your certificate. This means a certificate bundle that includes the public certificate of each Intermediate CA in the hierarchy of CAs that signed your certificate leading back to and including the Root CA. CAs typically provide such a bundle pre-configured, or you can create your own by downloading the public certificates of the CAs in your chain.

The full chain file is simply a certificate bundle that starts with your server certificate, followed by the same contents as the Root & Intermediate certificate bundle. It includes the public certificates of each step in the chain – from the server certificate, all the way to the Root CA. You may have to create this full chain file yourself depending on the options your CA provides.

Certificate bundles using PEM files are created by simply concatenating the text versions of the certificates together in a single file. The text version of a certificate in a PEM file is the encoded text in between and including the **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----** tag lines.

To create a bundle file:

a. Open the certificate file to include using a plain text editor such as Notepad or Text Edit. Highlight and copy the block of encoded certificate text including the **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----** lines.

b. Paste the certificate contents into a new empty text file.

c. For each additional certificate to add: open and copy the block of encoded certificate text including the BEGIN and END tag lines and paste at the end of the new text file that was created in Step . Each certificate should start on its own line, with no extra lines in between certificates. Certificates should be pasted in hierarchical order so that the file ends with the Root Certificate.

d. Place one extra blank line at the end of the file and save the text file with an extension of .pem .cer or .crt. Example: **single-chain.crt**

6. Using your SFTP client, log into Meeting Server and copy the signed certificate file, certificate authority bundle, and full chain file to your Meeting Server.

Note: File names are restricted on Meeting Server, so your files must use common file extensions such as **.crt**, **.cer**, **.key**, **.pem** or **.der**

D.3 Configuring Meeting Server to use Web Bridge 3

If upgrading your Meeting Server to 2.9, by default, this release will use your existing Web Bridge 2 configuration. However, you can configure Web Bridge 3 to operate at the same time as Web Bridge 2. Web Bridge 2 uses XMPP and Web Bridge 3 uses Call Bridge to Web Bridge (C2W) protocol connections so they can work in parallel, however, they will need to be configured to use different ports.

Note: You don't need to configure an XMPP Server for Web Bridge 3.

Web Bridge 3 is similar to Web Bridge 2 for configuration and setup which is done using MMP commands via SSH. The main difference is that Web Bridge 2 requires configuring an HTTPs port, whereas Web Bridge 3 requires configuring an HTTPS port and a C2W port.

To configure Meeting Server to use Web Bridge3:

1. SSH into the MMP and log in.
2. Use the **webbridge3** command in the MMP to configure webbridge3. To display the webbridge 3 usage, enter: **help webbridge3**

```
> help webbridge3

    Usage:
    webbridge3
    webbridge3 restart
    webbridge3 enable
    webbridge3 disable
    webbridge3 https listen <interface:port allowed list>
    webbridge3 https certs  <key-file> <crt-fullchain-file>
    webbridge3 https certs none
    webbridge3 http-redirect (enable [port]|disable)
    webbridge3 c2w listen <interface:port allowed list>
    webbridge3 c2w certs  <key-file> <crt-fullchain-file>
    webbridge3 c2w certs none
    webbridge3 c2w trust <crt-bundle>
    webbridge3 c2w trust none
    webbridge3 options <space-separated options>
    webbridge3 options none
    webbridge3 status
```

3. (Optional) Set up a port for HTTP connections. This port will be opened for all Meeting Server interfaces on which the web app has been configured. Incoming HTTP connections will be automatically redirected to the matching HTTPS port for the interface they arrived on. The default port, if you don't specify one in **webbridge3 http-redirect enable [port]**, is 80.
4. Configure the port for the HTTPS service to listen to. To configure it to listen on port 443 of the a interface:

   webbridge3 https listen a:443
5. Set the HTTPS certificates. These are the certificates that will be presented to web browsers so they need to be signed by a certification authority and the hostname/purpose etc needs to match. (The certificate file is the full chain of certificates that starts with the end entity certificate and finishes with the root certificate.) Enter the command:

   webbridge3 https certs wb3-https.key wb3-https-fullchain.crt
6. Configure the C2W connection. We recommend that you make this address/port accessible from the Call Bridge(s) only. The following command sets it in port 9999 of

interface a:

```
webbridge3 c2w listen a:9999
```

Note that here we use the example of port 9999, however, it can be any available port on your network. It's not a fixed port, unlike 443.

7. Configure the C2W connection certificates. You need to configure the SSL Server certificates used for the C2W connection. (See "Configuring Call bridge to use C2W connections" below for certificate requirements, and more information can be found in this FAQ.)

```
webbridge3 c2w certs wb3-c2w.key wb3-c2w-fullchain.crt
```

8. The Web Bridge 3 C2W server is expecting Call Bridges to present a client certificate – it will verify whether to trust them using the trust bundle provided by the following command:

```
webbridge3 c2w trust wb3-c2w-trust-bundle.crt
```

9. Now enable Web Bridge 3:

```
webbridge3 enable
```

D.4 Configuring Call bridge to use C2W connections

C2W certificates are used for the connection between Call Bridge and Web Bridge 3. For the Call Bridge to make a C2W connection to a Web Bridge 3, you need to specify a C2W trust store to verify certificates against, i.e. the ones presented by the Web Bridge 3 that were configured in step 7 above.

1. Use the **callbridge** command in the MMP to display the Call Bridge usage, enter: **help callbridge** to display:

```
> help callbridge
Configure CMS callbridge

Usage:

    callbridge listen <interface allowed list>
    callbridge prefer <interface>
    callbridge certs <key-file> <crt-file> [<cert-bundle>]
    callbridge certs none
    callbridge trust xmpp <bundle>
    callbridge trust xmpp none
    callbridge trust c2w <bundle>
    callbridge trust c2w none
    callbridge add edge <ip address>:<port>
    callbridge del edge
```

```
callbridge trust edge <trusted edge certificate bundle>
callbridge trust cluster none
callbridge trust cluster <trusted cluster certificate bundle>
callbridge restart
```

2. Set the certificates for the Call Bridge:

   ```
   callbridge certs cert.key cert.crt
   ```

3. Set the C2W trust store that will be used to validate the SSL Server certificate presented by the Web Bridge 3. (For more information, see this FAQ.)

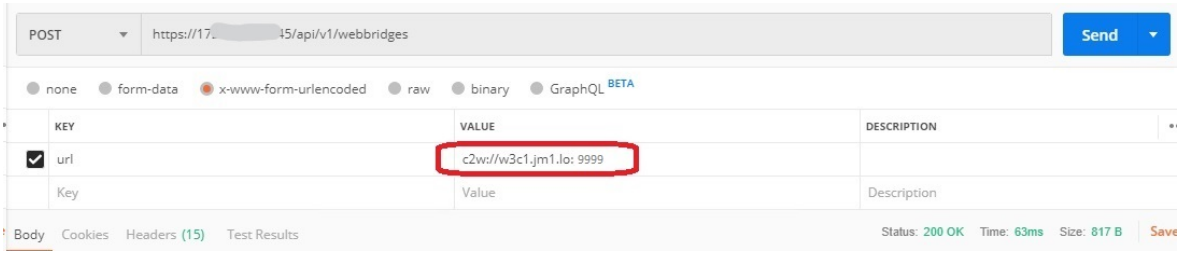
   ```
   callbridge trust c2w c2w-callbrige-trust-store.crt
   ```

4. Now restart Call Bridge:

   ```
   callbridge restart
   ```

5. Register the Web Bridge 3 URL to the running callbridge REST API in the same way as you would for Web Bridge 2, as shown below, i.e. POST to /api/v1/webbridges with the "url" parameter. The URL protocol indicates if it is webbridge2 or webbridge3. So, if the protocol is http:// or https:// then the webbridge is treated as webbridge2, and if you specify c2w:// protocol in the URL then it will be handled as a webbridge3 connection.

   Figure 4: Registering Web Bridge 3 URL to the Call Bridge API



Cisco Legal Information

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco Trademark