Cisco TelePresence Conductor Certificate Creation and Use

Deployment Guide

XC3.0

January 2015

Contents

Introduction

This deployment guide provides instructions on how to create X.509 cryptographic certificates for use with the Cisco TelePresence Conductor (TelePresence Conductor), and how to load them into TelePresence Conductor.

PKI introduction

Public Key Infrastructure (PKI) provides the mechanisms through which communications can be secured (encrypted and integrity protected) and identities can be verified. Underlying PKI is:

- **A public/private key pair**: a public key is used to encrypt data sent to a server, but only the private key (kept secret by the server) can be used to decrypt it.
- **Signatures of data**: data can be "signed" by a server, by using a combination of a cryptographic hash of the data and the server's private key. A client can verify the signature by using the server's public key and verifying the same hash. This ensures the data has been sent from the expected server, and has not been tampered with.
- **Certificates**: a certificate is a wrapper around a public key, and provides information about the owner of the key. This metadata is provided in X.509 format, and typically includes the server name and contact details for the owner.
- **A certificate chain**: a certificate can be signed by a Certificate Authority (CA) using its own private key. In turn, therefore, a certificate can be verified as being signed by a CA by checking the signature against the CA's certificate (public key). Web browsers and other clients have a list of CA certificates that they trust, and can thus verify the certificates of individual servers.

Transport Layer Security (TLS) is the standard mechanism for securing a TCP connection between hosts on a TCP/IP network. For example, secure HTTP (HTTPS) uses TLS to encrypt and verify traffic. To establish a TLS connection:

1. An initial TCP connection is made, and the client sends its capabilities (including cipher suites) and a random number.
2. The sever responds with its choice of those capabilities, another random number, and its certificate.
3. The client verifies that the server certificate was issued (signed) by a CA that it trusts, and has not been revoked.
4. The client sends a "pre-master secret", encrypted with the server's public key.
5. This pre-master secret, combined with the exchanged random numbers (to prevent replay attacks), is used to generate a "master secret", with which the remaining communications of this TLS session are encrypted between the client and server.

The following sections describe how these PKI components can be used with the TelePresence Conductor.

Overview of certificate use on the TelePresence Conductor

TelePresence Conductor needs certificates for:

- Secure HTTP with TLS (HTTPS) connectivity
- TLS connectivity for SIP signaling
- Connections to other systems such as Cisco TMS, Cisco VCS, Unified CM, TelePresence Server, TelePresence MCU, LDAP servers and syslog servers

It uses its list of trusted Certificate Authority (CA) certificates and associated certificate revocation lists (CRLs) to validate other devices connecting to it.

It uses the Server Certificate and the Private key to provide a signed certificate to provide evidence that the TelePresence Conductor is the device it says it is. This can be used with connected systems, as well as administrators using the web interface.

A certificate identifies the TelePresence Conductor. It contains names by which it is known and to which traffic is routed. If the TelePresence Conductor is known by multiple names for these purposes, such as if it is part of a cluster, this must be represented in the X.509 subject data, according to the guidance of RFC5922. The certificate must contain the FQDN of both the TelePresence Conductor itself and of the cluster. If a certificate is shared across cluster peers, it must list all possible peer FQDNs. The following lists show what must be included in the X.509 subject, depending on the deployment model chosen.

If the TelePresence Conductor is not clustered:

- Subject Common Name = FQDN of TelePresence Conductor
- Subject Alternate Names = leave blank

If the TelePresence Conductor is clustered, with individual certificates per TelePresence Conductor:

- Subject Common Name = FQDN of TelePresence Conductor
- Subject Alternate Names = FQDN of TelePresence Conductor, FQDN of cluster

Wildcard certificates manage multiple subdomains and the services names they support, they can be less secure than SAN (Subject Alternate Name) certificates. TelePresence Conductor does not support wildcard certificates.

Certificate generation overview

X.509 certificates may be supplied from a third party, or may be generated by a certificate generator such as OpenSSL or a tool available in applications such as Microsoft Certification Authority. Third-party certificates supplied by recognized certificate authorities are recommended, although TelePresence Conductor deployments in controlled or test environments can use internally generated certificates.

Certificate generation is usually a 3-stage process:

- Stage 1: generate a private key
- Stage 2: create a certificate request
- Stage 3: authorize and create the certificate

This document presents alternative methods of generating the root certificate, client/server certificate for the TelePresence Conductor, and private key:

- Generating a certificate signing request (CSR) [p.6] describes how to use the TelePresence Conductor itself to generate the private key and certificate request.
- Appendix 2: Certificate generation using OpenSSL only [p.13] documents the OpenSSL-only process, which could be used with a third party or internally managed CA.

For mutual TLS authentication the TelePresence Conductor **Server** certificate must be capable of being used as a **Client** certificate as well, thus allowing the TelePresence Conductor to authenticate as a client device to a neighboring server (see Appendix 5: Configuring Windows Server Manager with a "client and server" certificate template [p.22]).

Note: It is worth noting that changes are being introduced to the way that dates are handled from 2050, and certificates that have expiry dates beyond that can cause operational issues.

Generating a certificate signing request (CSR)

A CSR contains the identity information about the owner of a private key. It can be passed to a third-party or internal certification authority for generating a signed certificate, or it can be used in conjunction with an application such as Microsoft Certification Authority or OpenSSL.

Note: The TelePresence Conductor can accept and use certificates generated with SHA-256 hashing, but the CSR (certificate signing request) generator on the user interface does not provide the option to select SHA-256.

Creating a CSR using TelePresence Conductor

The TelePresence Conductor can generate server certificate signing requests. This removes the need to use an external mechanism to generate and obtain certificate requests.

To generate a CSR:

1. Go to **Maintenance > Security certificates > Server certificate**.
2. Click **Generate CSR** to go to the **Generate CSR** page.
3. Enter the required properties for the certificate.
   - See Server certificates and clustered systems [p.7] if your TelePresence Conductor is part of a cluster.
   - The certificate request includes automatically the public key that will be used in the certificate, and the client and server authentication Enhanced Key Usage (EKU) extension.
4. Click **Generate CSR**. The system will produce a signing request and an associated private key.
   The private key is stored securely on the TelePresence Conductor and cannot be viewed or downloaded. You must never disclose your private key, not even to the certificate authority.
5. You are returned to the **Server certificate** page. From here you can:
   - **Download** the request to your local file system so that it can be sent to a certificate authority. You are prompted to save the file (the exact wording depends on your browser).
   - **View** the current request.

Note:

- Only one signing request can be in progress at any one time. This is because the TelePresence Conductor has to keep track of the private key file associated with the current request. To discard the current request and start a new request, click **Discard CSR**.
- The certificate signing request storage location changed in XC3.x.
  When you generate a CSR in XC2.x, the application puts **csr.pem** and **privkey_csr.pem** into **/tandberg/persistent/certs**.
  When you generate a CSR in XC3.x, the application puts **csr.pem** and **privkey.pem** into **/tandberg/persistent/certs/generated_csr**.
  If you want to upgrade from XC2.x and have an unsubmitted CSR, then we recommend discarding the CSR before upgrade, and then regenerating the CSR after upgrade.

You must now authorize the request and generate a signed PEM certificate file. You can pass it to a third-party or internal certification authority, or use it in conjunction with an application such as Microsoft Certification Authority (see Authorizing a request and generating a certificate using Microsoft Certification Authority [p.8]) or OpenSSL (see Operating as a Certificate Authority using OpenSSL [p.15]).

When the signed server certificate is received back from the certificate authority, it must be uploaded to the TelePresence Conductor as described in Loading certificates and keys onto TelePresence Conductor [p.10].

Server certificates and clustered systems

When a CSR is generated, a single request and private key combination is generated for that peer only.

If you have a cluster of TelePresence Conductors, you must generate a separate signing request on each peer. Those requests must then be sent to the certificate authority and the returned server certificates uploaded to each relevant peer.

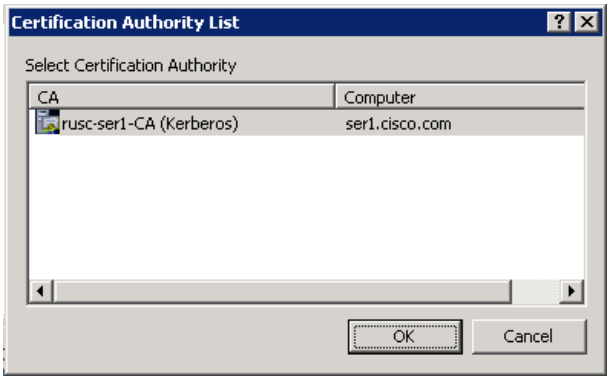
You must ensure that the correct server certificate is uploaded to the appropriate peer, otherwise the stored private key on each peer will not correspond to the uploaded certificate.

Authorizing a request and generating a certificate using Microsoft Certification Authority

This section describes how to authorize a certificate request and generate a PEM certificate file using Microsoft Certification Authority.

1. Copy the certificate request file (for example, **certcsr.der** if generated via OpenSSL) to a location, such as the desktop, on the server where the Microsoft Certification Authority application is installed.
2. Submit the certificate request from a command prompt:
   - To generate a certificate with Server Authentication and Client Authentication type:
     `certreq -submit -attrib "CertificateTemplate:Webclientandserver" C:\Users\<user>\Desktop\certcsr.der`
     See Appendix 5: Configuring Windows Server Manager with a "client and server" certificate template [p.22] for details about how to set up the **`Webclientandserver`** certificate template.
   - To generate a certificate with Server Authentication only, type:
     `certreq -submit -attrib "CertificateTemplate:WebServer" C:\Users\<user>\Desktop\certcsr.der`

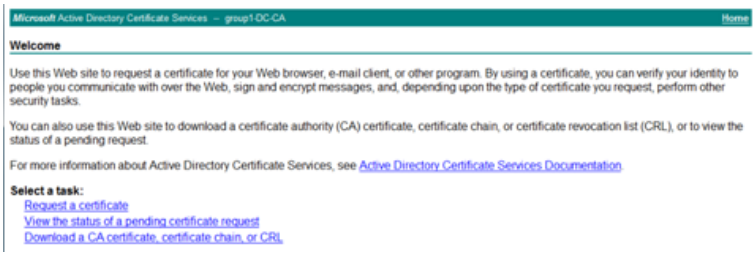
   This triggers the Certification Authority window to open:



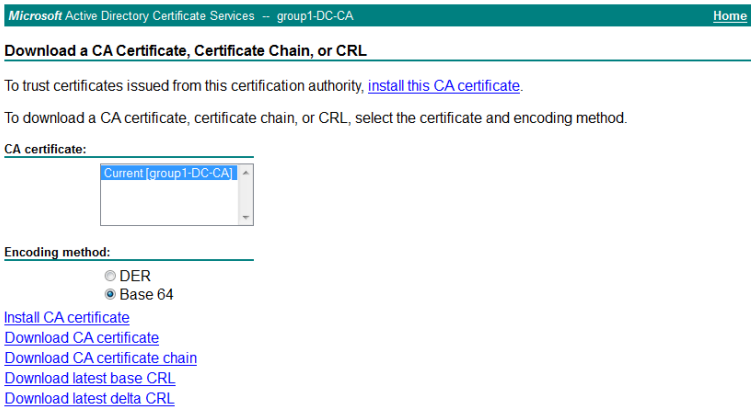
   Note that the command must be run as the administrator user.
3. Select the **Certification Authority** to use (typically only one is offered) and click **OK**.
4. When requested, save the certificate (browse to the required folder if the default **Libraries > Documents** folder is not to be used) calling it **server.cer** for example.
5. Rename **server.cer** to **server.pem** for use with the TelePresence Conductor.

Get the Microsoft CA certificate

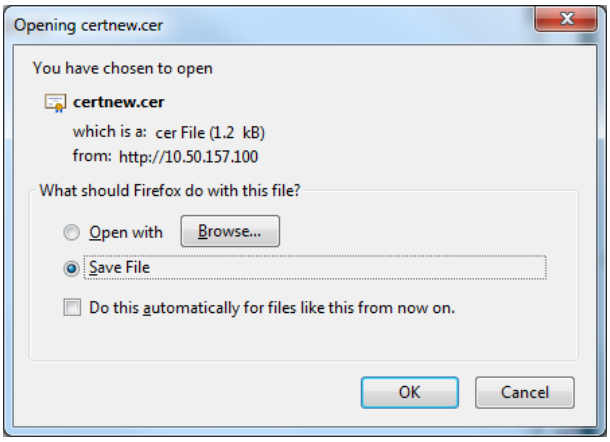
1. In your web browser, go to <IP or URL of the Microsoft Certificate Server>/certsrv and log in.



2. Select **Download a CA certificate, certificate chain or CRL**.



3. Select **Base 64**.
4. Select **Download CA certificate**.



5. Choose **Save File** and click **OK**.
6. Rename **certnew.cer** to **certnew.pem**.

Files **server.pem** and **certnew.pem** are now available.

Go to the Loading certificates and keys onto TelePresence Conductor [p.10] section in this document and upload **server.pem** and **certnew.pem** to TelePresence Conductor.

Loading certificates and keys onto TelePresence Conductor

The TelePresence Conductor uses standard X.509 certificates. The certificate information must be supplied to the TelePresence Conductor in PEM format. Typically 3 elements are loaded:

- The server certificate (which is generated by the certificate authority, identifying the ID of the certificate holder, and should be able to act as both a client and server certificate).
- The private key (used to sign data sent to the client, and decrypt data sent from the client, encrypted with the public key in the server certificate). This must only be kept on the TelePresence Conductor and backed up in a safe place – security of the TLS communications relies upon this being kept secret.
- A list of certificates of trusted certificate authorities.

Note: New installations of TelePresence Conductor software (from XC2.3 onwards) ship with a temporary trusted CA, and a server certificate issued by that temporary CA. We strongly recommend that you replace the server certificate with one generated by a trusted certificate authority, and that you install CA certificates for the authorities that you trust.

Loading a server certificate and private key onto TelePresence Conductor

The TelePresence Conductor's server certificate is used to identify the TelePresence Conductor when it communicates with client systems using TLS encryption, and with web browsers over HTTPS.

To upload a server certificate:

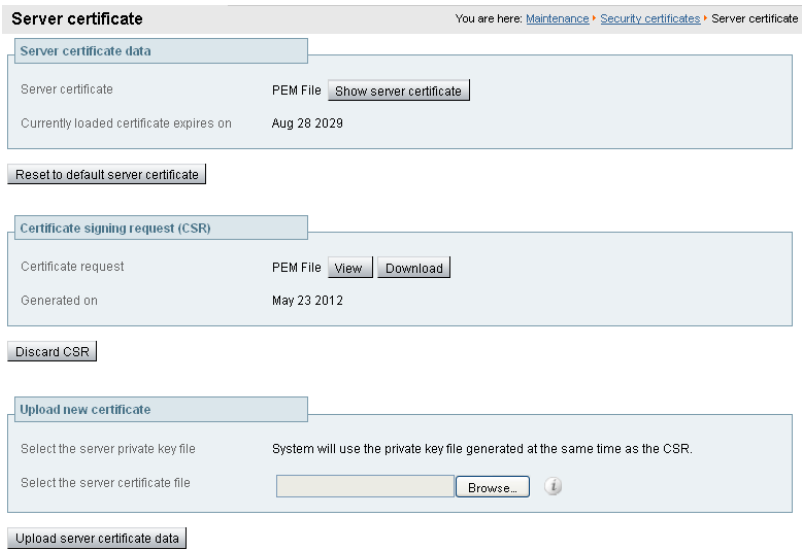
1. Go to **Maintenance > Security certificates > Server certificate**.
2. Use the **Browse** button in the **Upload new certificate** section to select and upload the **server certificate** PEM file.
3. If you used an external system to generate the Certificate Signing Request (CSR) you must also upload the **server private key** PEM file that was used to encrypt the server certificate. (The private key file will have been automatically generated and stored earlier if the TelePresence Conductor was used to produce the CSR for this server certificate.)
   - The **server private key** PEM file must not be password protected.
   - You cannot upload a server private key if a certificate signing request is in progress.
4. Click **Upload server certificate data**.

The certificate signing request storage location changed in XC3.x.

When you generate a CSR in XC2.x, the application puts **csr.pem** and **privkey_csr.pem** into **/tandberg/persistent/certs**.

When you generate a CSR in XC3.x, the application puts **csr.pem** and **privkey.pem** into **/tandberg/persistent/certs/generated_csr**.

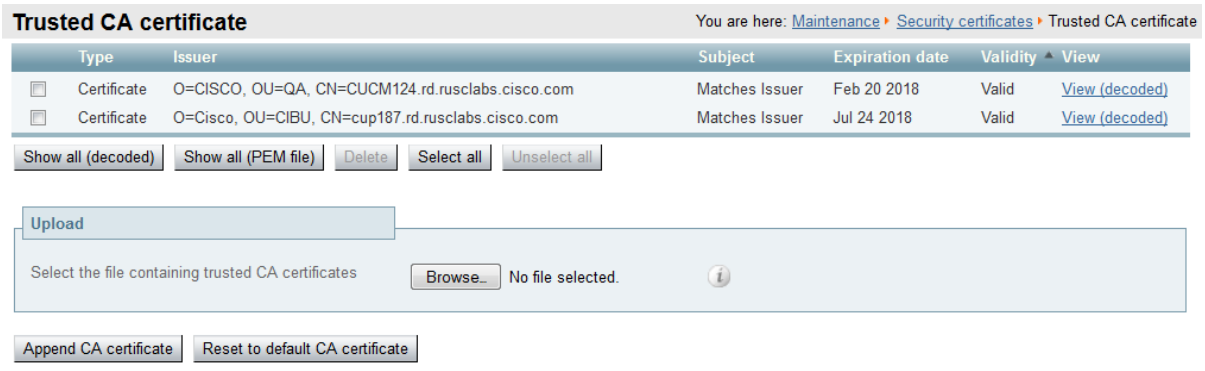
If you want to upgrade from XC2.x and have an unsubmitted CSR, then we recommend discarding the CSR before upgrade, and then regenerating the CSR after upgrade.



Managing the trusted CA certificate list

The **Trusted CA certificate** page (**Maintenance > Security certificates > Trusted CA certificate**) allows you to manage the list of certificates for the Certificate Authorities (CAs) trusted by this TelePresence Conductor. When a TLS connection to TelePresence Conductor mandates certificate verification, the certificate presented to the TelePresence Conductor must be signed by a trusted CA in this list and there must be a full chain of trust (intermediate CAs) to the root CA.

- To upload a new file containing one or more CA certificates, **Browse** to the required PEM file and click **Append CA certificate**. This will append any new certificates to the existing list of CA certificates. If you are replacing existing certificates for a particular issuer and subject, you have to manually delete the previous certificates.
- To replace all of the currently uploaded CA certificates with the system's original list of trusted CA certificates, click **Reset to default CA certificate**.
- To view the entire list of currently uploaded trusted CA certificates, click **Show all (decoded)** to view it in a human-readable form, or click **Show all (PEM file)** to view the file in its raw format.
- To view an individual trusted CA certificate, click on **View (decoded)** in the row for the specific CA certificate.
- To delete one or more CA certificates, tick the box(es) next to the relevant CA certificate(s) and click **Delete**.



Appendix 1: Troubleshooting

Certificates with key length of 8192 bits

Problems have been seen with some 3rd Party web browsers when using 8192 bit keys. We recommend using certificates with a key length of 4096 bits.

Issues with ssh failures and unsupported OIDs

If you experience unknown ssh failures such as ssh tunnels failing to establish, please verify there are no unknown OIDs in the certificate. This can be done by checking that there are no undecoded numerical entries in the CN of the Issuer & Subject fields (from the GUI: **Maintenance -> Security Certificates -> Server Certificate -> Show(decoded)** or from the console: '**`openssl x509 -text -noout -in /tandberg/persistent/certs/server.pem`**')

```
Invalid

subject=CN=blahdeblah,OU=IT

Security,O=BigBang,L=Washington,ST=District of

Columbia,C=US,1.3.6.1.4.1.6449.1.2.1.5.1 = #060C2B06010401B2310102010501

Valid

subject=CN=blahdeblah,OU=IT

Security,O=BigBang,L=Washington,ST=District of

Columbia,C=US,jurisdictionOfIncorporationLocalityName=Dover
```

For instance, currently, the only supported Extended Validation OIDs are:

- 1.3.6.1.4.1.311.60.2.1.1 jurisdictionOfIncorporationLocalityName
- 1.3.6.1.4.1.311.60.2.1.2 jurisdictionOfIncorporationStateOrProvinceName
- 1.3.6.1.4.1.311.60.2.1.3 jurisdictionOfIncorporationCountryName.

Appendix 2: Certificate generation using OpenSSL only

This section describes the process for generating a private key and certificate request for the TelePresence Conductor using OpenSSL. This is a generic process that relies only on the free OpenSSL package and not on any other software. It is appropriate when certificates are required for interfacing with neighboring devices for test purposes, and for providing output to interact with Certificate Authorities.

Note: The TelePresence Conductor can accept and use certificates generated with SHA-256 hashing, but the CSR (certificate signing request) generator on the user interface does not provide the option to select SHA-256.

The output for the certificate request generation process can be given to a Certificate Authority which may be internal or external to the organization, and which can be used to produce the X.509 certificates required by the TelePresence Conductor to authenticate itself with neighboring devices.

This section also briefly describes how OpenSSL could be used to manage a private Certificate Authority, but does not intend to be comprehensive. Various components of these processes can be used when interfacing with third party CAs.

OpenSSL and Mac OS X or Linux

OpenSSL is already installed on Mac OS X, and is usually installed on Linux.

OpenSSL and Windows

If you do not have OpenSSL already installed, this is available as a free download from http://www.openssl.org/related/binaries.html.

Choose the relevant 32 bit or 64 bit OpenSSL - the 'Light' version is all that is needed.

If you receive a warning while installing OpenSSL that C++ files cannot be found, load the "Visual C++ Redistributables" also available on this site and then re-load the OpenSSL software.

Creating a certificate request using OpenSSL

This process creates a private key and certificate request for the server that can then be validated by a CA. This could be a CA that has been created and managed locally, or a third-party CA.

Note: This method to create a CSR should only be used if you have a good knowledge of working with OpenSSL as there is a potential for entering incorrect commands (especially with numerous SAN entries). Missing relevant SAN entries would require recreating the certificate at a later date.

To generate the CSR from the command line with OpenSSL use these instructions:

1. SSH to the TelePresence Conductor and log in as root.
2. Make a new directory to do the work in - **`mkdir /tmp/certtemp`**
3. Move in to this directory - **`cd /tmp/certtemp`**
4. Copy the Open SSL configuration file we use for CSR to this directory, as we need to edit it (**Note: Keep the dot at the end**) - **`cp /etc/openssl/csrreq.cnf .`**
5. Open the file for editing – **`vi csrreq.cnf`**

6. Find the line "**default_md = sha1**" and edit it so that it reads "**default_md = sha256**"
7. Uncomment the line "**# req_extensions = v3_req**" by removing the **#** at the start of it
8. Make sure that the line "**extendedKeyUsage=serverAuth, clientAuth**" is present within the section [**v3_req**].
9. Find the line "**subjectAltName = ${ENV::CSR_ALT_NAME}**" and replace it such that it lists what you want in the Subject Alternative Names in the certificate e.g. "**subjectAltName = DNS:peer1vcs.example.com,DNS:peer2vcs.example.com,DNS:ClusterFQDN.example.com**". Make sure you add all the additional relevant entries. For MRA this may comprise:
   a. **Expressway E:** **DNS:<CM domain name>, DNS:<XMPP federation domain>, DNS:<federation chat alias 1>, DNS:<federation chat alias 2>**, etc.
   b. **Expressway C:** **DNS:<secure profile name 1>, DNS:<secure profile name 2>**, etc.
10. Now save the file and exit.
11. Run the following OpenSSL command to generate a new CSR and Private key for the VCS "**openssl req -nodes -newkey rsa:4096 -keyout privatekey.pem -out myrequest.csr -config csrreq.cnf**" changing the **rsa:nnnn** if required. (nnnn = keylength, recommended number is **4096**).
12. On the screen you will get output similar to what follows, some things can & should be left blank. When this is complete, there will be two new files, **myrequest.csr** and **privatekey.pem**.The required fields that should be completed are:
    - Country
    - State and province
    - Locality name
    - Organization name
    - Common name - this is the TelePresence Conductor cluster FQDN if the certificate is for a cluster of TelePresence Conductors or it is the FQDN of the TelePresence Conductor if the certificate is for a single TelePresence Conductor
    - Email address - optional, can leave blank
    - A challenge password - optional, can leave blank
    - An optional company name - optional, can leave blank

```
Generating a 4096 bit RSA private key
............++
.....................................................++
writing new private key to 'privatekey.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a
DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:GB
State or Province Name (full name) [Some-State]:Berkshire
Locality Name (eg, city) []:Reading
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Cisco
Organizational Unit Name (eg, section) []:CIBU
Common Name (eg, YOUR name) []:rusc01-et-xm030.rusclabs.cisco.com
```

```
Email Address []:
```

13. (Optional) If you want to validate the DNS entries have been entered correctly into the request, the **myrequest.csr** file can be decoded using the command: **`openssl req -text -noout -in myrequest.csr`**
14. Submit the CSR to your chosen Certificate Authority, who will provide the public certificate.
15. Upload the public certificate to the VCS via the **Maintenance > Security certificates > Server certificate** webpage, "**Select the server certificate file**" entry box.
16. Upload the **privatekey.pem** to the VCS via the **Maintenance > Security certificates > Server certificate** webpage, "**Select the server private key file**" entry box.

The **privatekey.pem** should be kept safe.

Operating as a Certificate Authority using OpenSSL

A major deployment is likely to make use of a third-party certificate authority, or already have one internal to an organization's IT department. However, you can use OpenSSL to manage certificates in a private certificate authority as outlined below.

If you have already configured OpenSSL to act as a CA, go to section Creating a signed certificate using OpenSSL [p.16].

Configuring OpenSSL to act as a CA

OpenSSL is powerful software, and when operating as a CA, requires a number of directories and databases to be configured for tracking issued certificates.

The list of directories and files can be found in the openssl configuration file under the section **`[ CA_default ]`**. By default, the files/directories required to be created are:

- A **demoCA** directory in the current directory, with 3 subdirectories **certs**, **newcerts** and **private**.
- An empty file called **index.txt** in the **demoCA** directory.
- A file called **serial** in the **demoCA** directory, storing a 2-digit number, such as "10".

For example, use the commands:

```
mkdir demoCA
cd demoCA
mkdir certs
mkdir newcerts
mkdir private
touch index.txt
echo 10 > serial
```

Creating a Certificate Authority using OpenSSL

This process creates a private key and certificate of a Certificate Authority (CA), which can then be used to validate other certificates. Note that this will not be trusted by devices outside of those on which it is explicitly installed.

From a command prompt:

1. Ensure that you are in the **demoCA** directory.
2. For Windows: copy **openssl.cfg** from the directory where OpenSSL is installed to the **demoCA** directory and rename it as **openssl_local.cfg**.
   For Mac OS X: copy **/System/Library/OpenSSL/openssl.cnf** to the **demoCA** directory and rename it as **openssl_local.cfg**.
3. Use a text editor to edit the **openssl_local.cfg** file that was created by the above copy command. Make the following modifications to the `[CA_default]` section:
   a. Ensure that the line `copy_extensions = copy` does not have a # at the beginning of the line. Delete the # if it is there. If the line remains commented out, it will strip attributes in the CSR and the SSL Server and SSL Client attributes will not appear in the certificate.
   b. Change `policy = policy_match` to `policy = policy_anything`
   c. Change `dir = ./demoCA` to `dir = .`
   d. Optionally, change `default_days = 365` (1 year validity of the generated certificate) to `default_days = 3650` (10 years, or choose another suitable value).
   e. Save the file.
4. Generate a private key for the CA by running the following command:
   `openssl genrsa -aes256 -out private/cakey.pem 4096`
   This will prompt for a password with which to encrypt the private key: choose a strong password and record it in a safe place. The cakey.pem file will be used to create the CA certificate and to sign other certificates and must also be kept secure.
5. Generate the CA certificate by running the following command.
   For Windows: `openssl req -new -x509 -days 3650 -key private/cakey.pem -config openssl_local.cfg -sha1 -extensions v3_ca -out cacert.pem`
   For OS X: `openssl req -new -x509 -days 3650 -key private/cakey.pem -config openssl_local.cfg -sha1 -extensions v3_ca -out cacert.pem`
6. Enter a passphrase for the key, and then enter the data requested, including:
   - Country
   - State or province
   - Locality name
   - Organization name
   - Organizational unit
   - Common name – this is typically the name of a contact person for this CA
   - Email address – optional, can leave blank

After entering the requested data, the operation completes and the certificate authority certificate **cacert.pem** is now available.

Creating a signed certificate using OpenSSL

This process signs the server certificate with the generated CA key, using the previously generated certificate request.

From a command prompt:

1. Ensure that you are in the **demoCA** directory.
2. Ensure that the certificate request file (**certcsr.pem**) is available:
   - If the certificate request was created using the TelePresence Conductor (recommended process): Copy the file downloaded from the TelePresence Conductor into the **demoCA** directory and rename it as **certcsr.pem**.
   - If the certificate request was created using OpenSSL:

Copy the previously generated certificate request into the **demoCA** directory and then covert it to PEM format by running the following command:
```
openssl req -in certcsr.der -inform DER -out certcsr.pem -outform PEM
```

3. Generate a signed server certificate by running the following command:
   ```
   openssl ca -config openssl_local.cfg -cert cacert.pem -keyfile
   private/cakey.pem -in certcsr.pem -out certs/server.pem -md sha1
   ```
   If you receive a "failed to update database TXT_DB error number 2" error message, you can remove the contents of the **index.txt** file and then rerun the command.
4. You will be prompted to enter the password for the CA's private key.

The signed certificate for the server is now available as **demoCA/certs/server.pem**.

Creating self-signed certificates using OpenSSL

We do not recommend creating self-signed certificates. They will not work in Unified Communications deployments.

Instead you should create a Certificate Authority using OpenSSL as described above.

Appendix 3: Converting a DER certificate file to PEM format

A private key, root (CA) certificate and the server / client certificate can be generated using third-party tools (or purchased from a certificate authority), and may be generated as PEM (required format, extension .pem) or DER (extension .cer) format files.

Certificates must be in PEM format for use on the TelePresence Conductor. Conversion from DER to PEM format can be done in one of two ways, either using OpenSSL or Windows, as documented in the following sections.

Converting a DER certificate file to a PEM file using OpenSSL

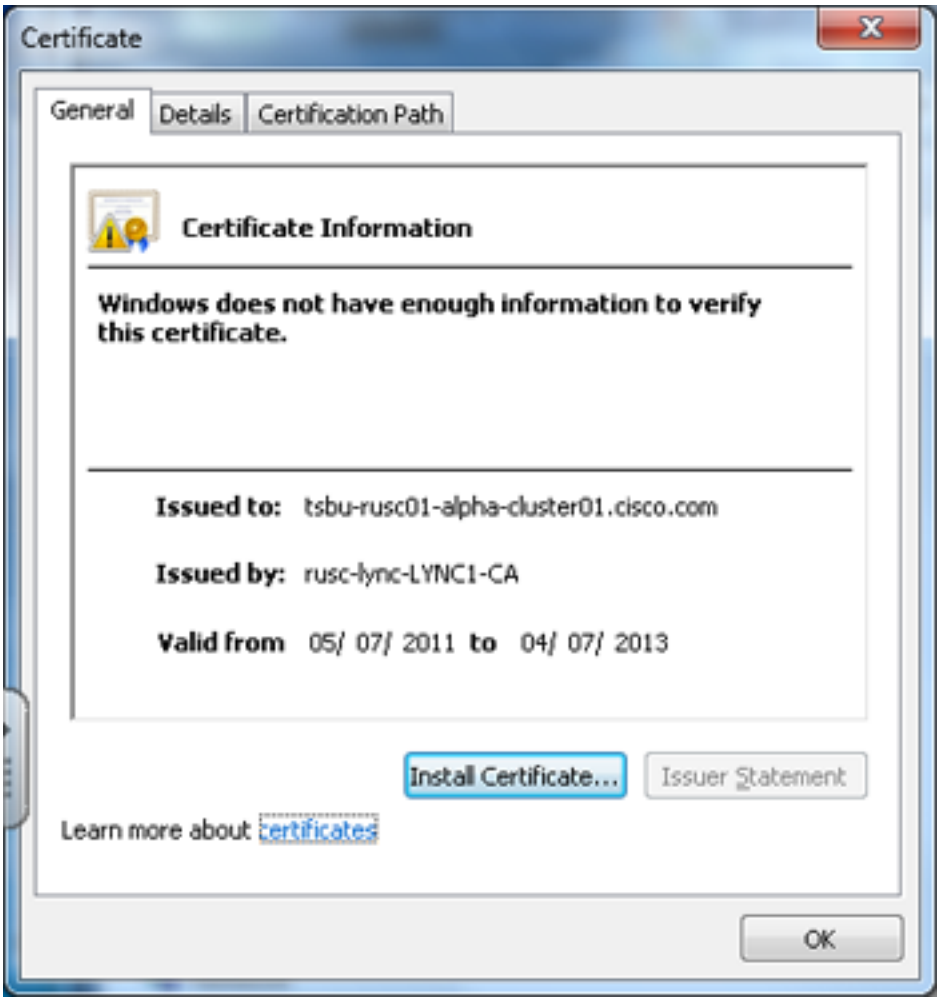
To convert from DER to PEM format, on a system running openssl, execute the command:

```
openssl x509 -in <filename>.cer -inform DER -out <filename>.pem -outform PEM
```

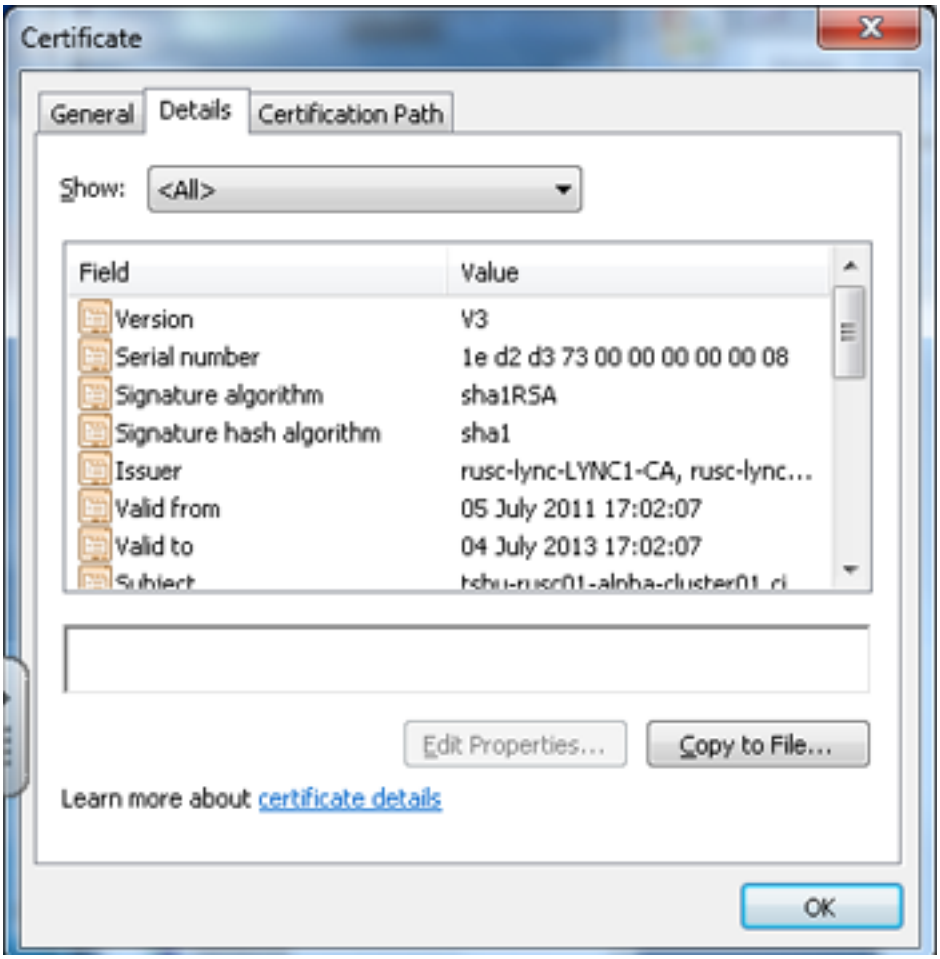
Converting a DER certificate file to a PEM file using Microsoft Windows

To convert from DER to PEM format using Microsoft Windows:

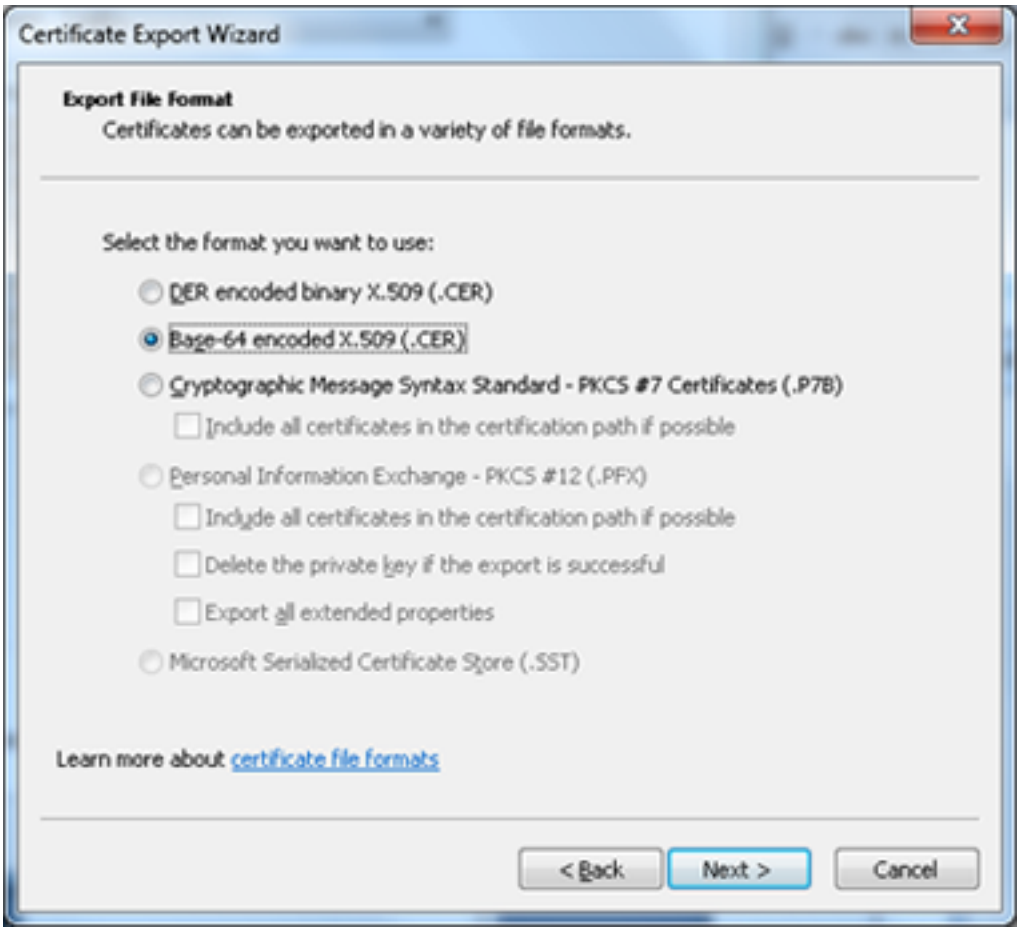
1. Double click on the DER file to convert (this will likely have a '.cer' extension).



2. Select the **Details** tab.



3. Click **Copy to File…**
4. On the **Welcome** page, click **Next**.
5. Select *Base-64 encoded X.509 (.CER)* and click **Next**.



6. Click **Browse** and select required destination for file (e.g. **server.pem**) and then click **Next**.
7. Click **Finish**.
8. Change the filename from **server.pem.cer** to **server.pem**.
9. This will be used in the Loading certificates and keys onto TelePresence Conductor [p.10] section of this document.

Appendix 4: Decoding certificates

This section describes some methods for decoding and viewing the content of certificates.

OpenSSL

A PEM file (e.g. **cert.pem**) can be decoded by the following command:

```
openssl x509 -text -in cert.pem
```

A DER file (e.g. **cert.cer**) can be decoded by the following command:

```
openssl x509 -text –inform DER -in cert.cer
```

Firefox

The certificate in use for a website being visited can be viewed in Firefox by clicking on the security information button on the address bar, and then clicking **More Information** followed by **View Certificate**.

Internet Explorer

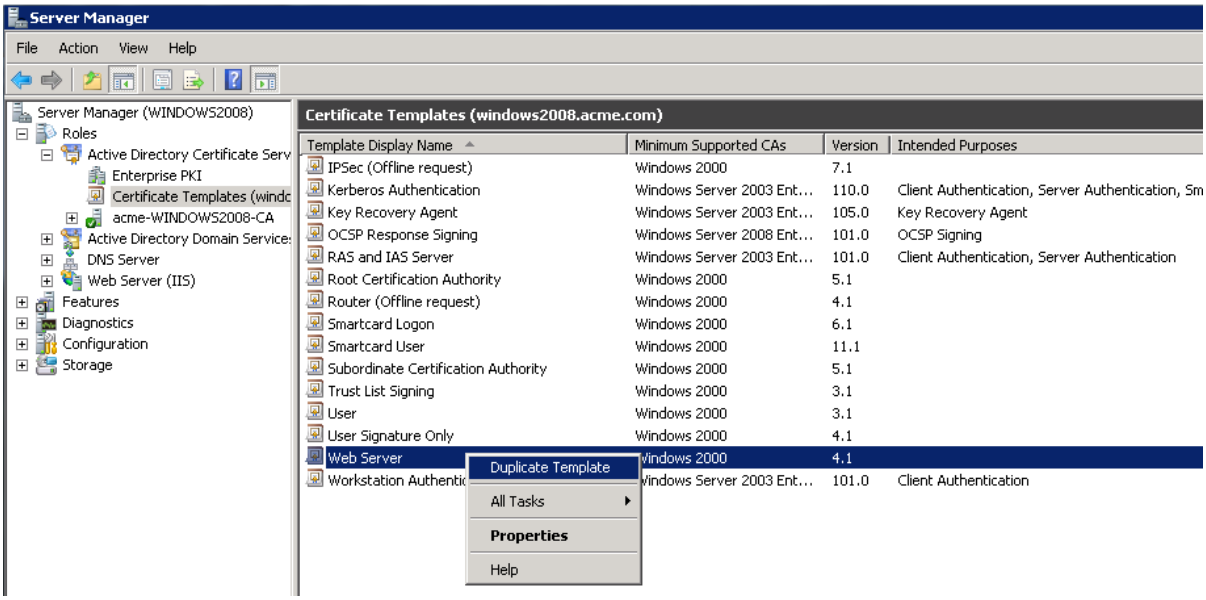
The certificate in use for a website being visited can be viewed in Internet Explorer by clicking the lock icon to the right of the address bar. A **Website Identification** dialog will appear. Click the **View Certificates** link at the bottom.

Appendix 5: Configuring Windows Server Manager with a "client and server" certificate template

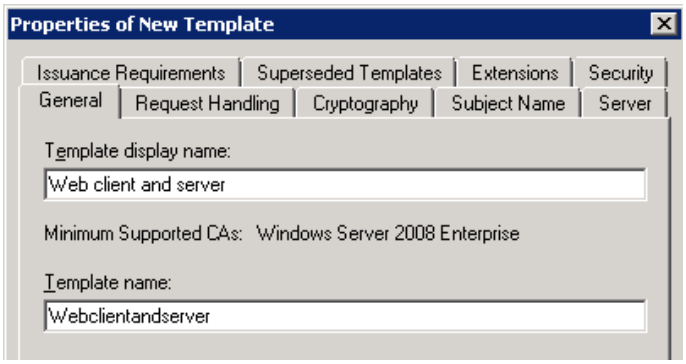
The default "Web Server" certificate template used by the Microsoft Certification Authority application will only create a certificate for Server Authentication. The server certificate for the TelePresence Conductor also needs Client Authentication.

To set up a certificate template with Server and Client Authentication:

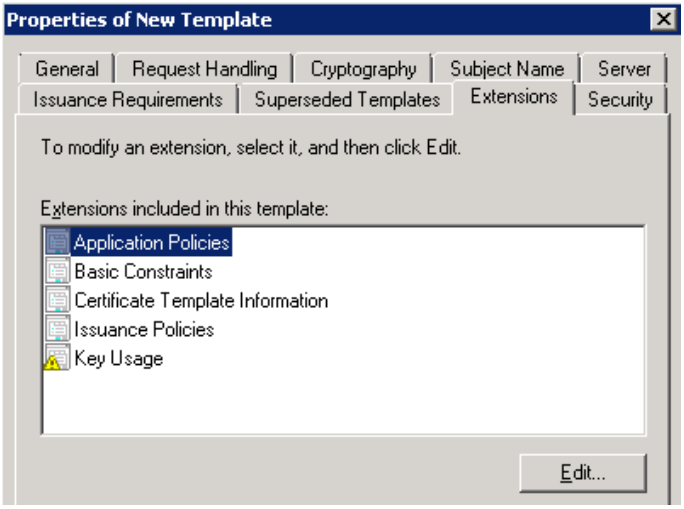
1. In Windows, launch **Server Manager** (**Start > Administrative Tools > Server Manager**). (Server Manager is a feature included with server editions of Windows.)
2. Expand the **Server Manager** navigation tree to **Roles > Active Directory Certificate Services > Certificate Templates (<domain>)**.
3. Right-click on **Web Server** and select **Duplicate Template**.



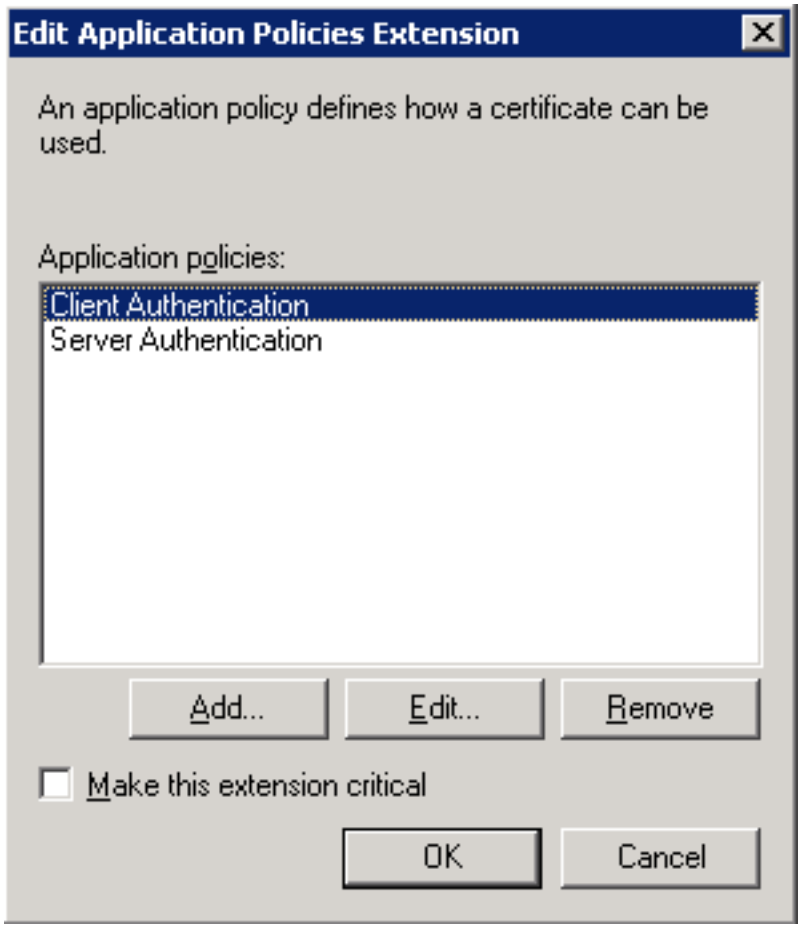
4. Select **Windows Server 2003 Enterprise** and click **OK**.
5. On the **General** tab, enter the **Template display name** and **Template name**, for example **`Web client and server`** and **`Webclientandserver`**.



6. On the **Extensions** tab, select **Application Policies** and click **Edit**.

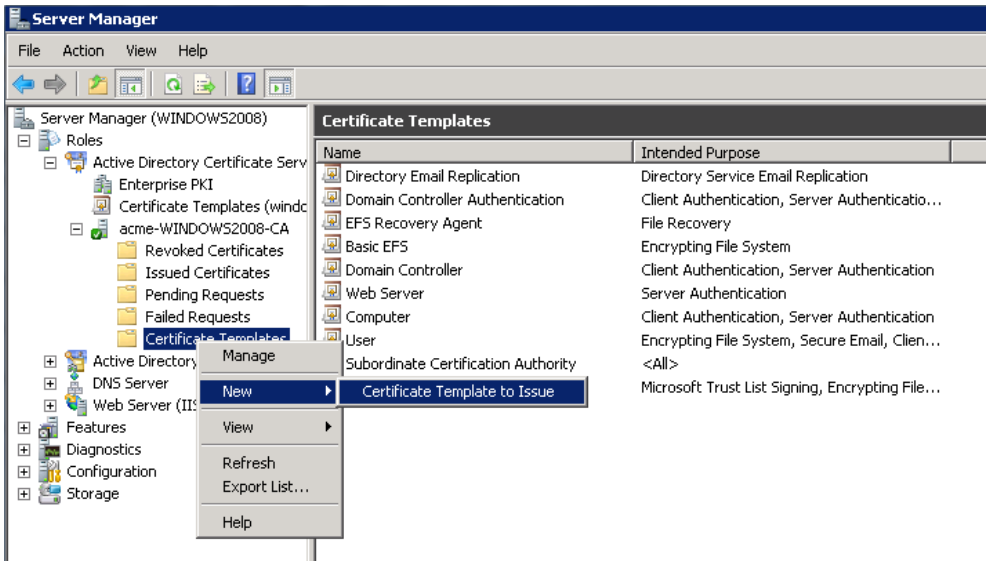


7. Add **Client Authentication** to the set of application policies:
   a. Click **Add**.
   b. Select **Client Authentication** and click **OK**.
   c. Click **OK**.

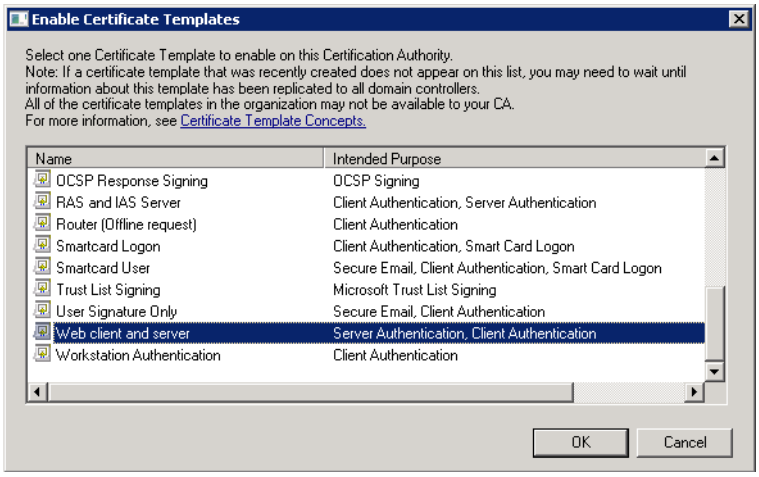


8. Click **OK** to complete the addition of the new template.

9. Add the new template to the Certificate Authority:
   a. Go to **Roles > Active Directory Certificate Services > <your certificate authority>**.
   b. Right-click on **Certificate Templates** and select **New > Certificate Template to Issue**.



   c. Select your new **Web client and server** template and click **OK**.



The new **Web client and server** template can now be used when submitting a certificate request to that Microsoft Certification Authority.

Document revision history

The following table summarizes the changes that have been applied to this document.

Revision	Date	Description
4	January 2015	Republished for XC3.0
3	August 2014	Republished for XC2.4.
2	April 2014	Republished for XC2.3. Removed "Certificate generation using Microsoft OCS" appendix. Various improvements and clarifications to "Certificate generation using OpenSSL only" appendix.
1	August 2013	Initial release.