Cisco TelePresence Certificate Creation and Use With Cisco VCS

Deployment Guide

Cisco VCS X7.2

D14548.07

February 2013

Contents

Introduction

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

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 The client verifies that the server certificate was issued (signed) by a CA that it trusts, and has not been revoked.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 VCS.

Overview of certificate use on the VCS

VCS needs certificates for:

- Secure HTTP with TLS (HTTPS) connectivity
- TLS connectivity for SIP signaling, endpoints and neighbor zones
- Connections to other systems such as Cisco TMS, 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 VCS is the device it says it is. This can be used with neighboring devices such as Microsoft Lync or Cisco Unified Communications Manager, as well as administrators using the web interface.

A certificate identifies the VCS. It contains names by which it is known and to which traffic is routed. If a VCS 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 VCS 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 VCS is not clustered:

- Subject Common Name = FQDN of VCS
- Subject Alternate Names = leave blank

If the VCS is clustered, with individual certificates per VCS:

- Subject Common Name = FQDN of VCS
- Subject Alternate Names = FQDN of VCS, FQDN of cluster

If the VCS is clustered, with a single certificate per cluster:

- Subject Common Name = FQDN of cluster
- Subject Alternate Names = FQDN of cluster, FQDN of VCS peer 1, …. FQDN of VCS peer n

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 Office Communications Server (OCS). Third-party certificates supplied by recognized certificate authorities are recommended, although VCS 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 VCS, and private key:

- Generating a certificate signing request (CSR) [p.5] describes how to use the VCS 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 VCS **Server** certificate must be capable of being used as a **Client** certificate as well, thus allowing the VCS to authenticate as a client device to a neighboring server (see Appendix 7 – Configuring Windows Server Manager with a "client and server" certificate template [p.32]).

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.

Creating a CSR using VCS (X7.2 or later)

The VCS 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 > Certificate management > 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.5] below if your VCS 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. Note that the private key is stored securely on the VCS and cannot be viewed or downloaded.
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 that only one signing request can be in progress at any one time. This is because the VCS 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**.

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.6]) or OpenSSL (see Appendix 2 – Certificate generation using OpenSSL only [p.13]).

When the signed server certificate is received back from the certificate authority, it must be uploaded to the VCS as described in Loading certificates and keys onto VCS [p.8].

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 VCSs, 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.

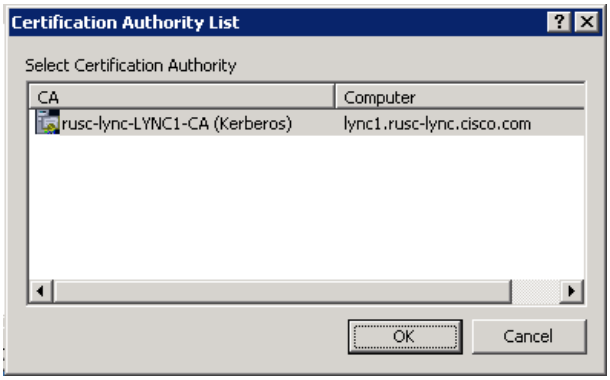
You must ensure that the correct server certificate is uploaded to the appropriate peer, otherwise the stored private key on each peer will not correspond to the uploaded certificate.

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. The Microsoft Certification Authority application may be installed on the Lync server, or another server in the network.

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, which is required if you want to configure a neighbor or traversal zone with mutual authentication (**TLS verify mode**), type:
     `certreq -submit -attrib "CertificateTemplate:Webclientandserver" C:\Users\<user>\Desktop\certcsr.der`
     See Appendix 7 – Configuring Windows Server Manager with a "client and server" certificate template [p.32] 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`

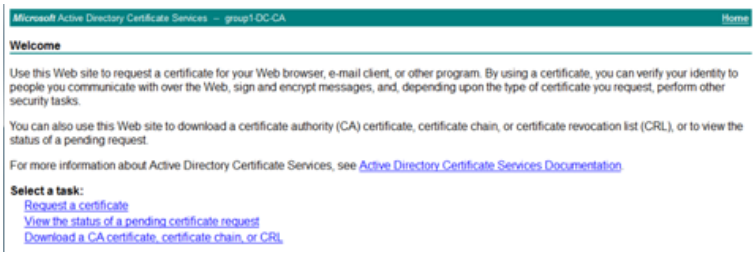
   This triggers the Certification Authority window to open:

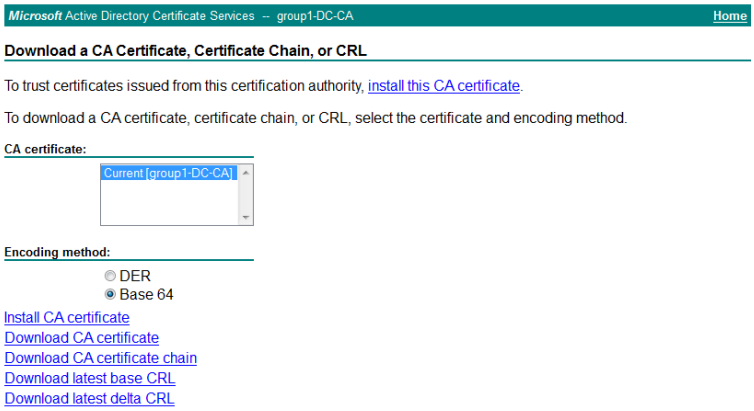
   Note that the command must be run as the administrator user.
3. Select the **Certification Authority** to use (typically only one is offered) and click **OK**.
4. When requested, save the certificate (browse to the required folder if the default **Libraries > Documents** folder is not to be used) calling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

Get the Microsoft CA certificate

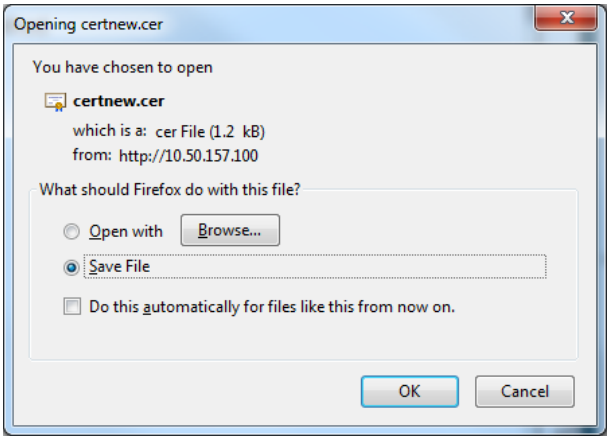
1. In your web browser, go to <IP or URL of the Microsoft Certificate Server>/certsrv and log in.

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 VCS [p.8] section in this document and upload **server.pem** and **certnew.pem** to VCS.

Loading certificates and keys onto VCS

The VCS uses standard X.509 certificates. The certificate information must be supplied to the VCS 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 VCS 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.

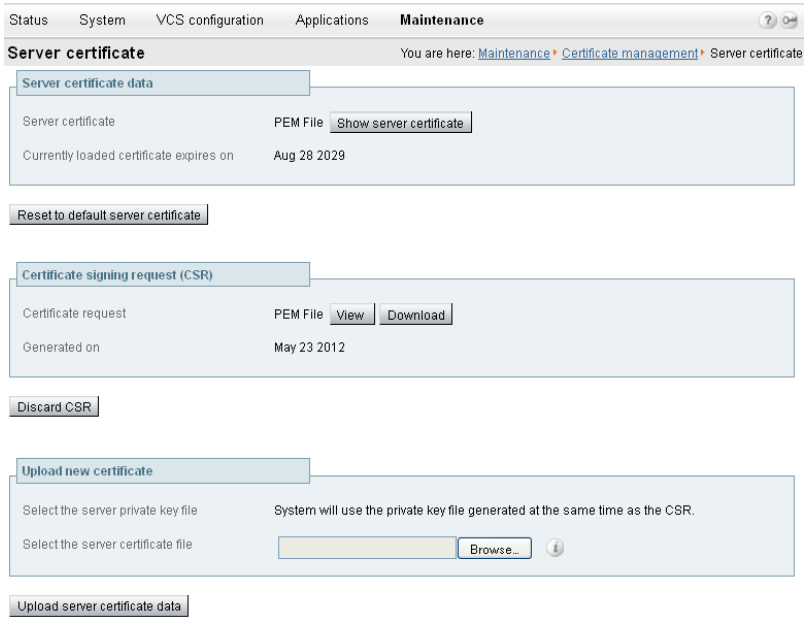
Loading a server certificate and private key onto VCS

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

Note: the VCS has a pre-loaded default server certificate. This is intended to be temporary and we strongly recommend that you replace it with your own certificate.

To upload a server certificate:

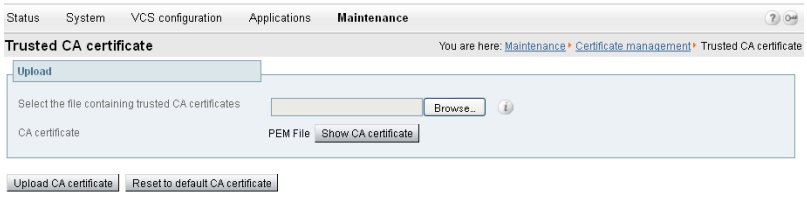
1. Go to **Maintenance > Certificate management > Server certificate**.
2. Use the **Browse** button to select and upload the **server certificate** PEM file.
3. If you used an external system to generate the certificate request you must also upload the **server private key** PEM file that was used to encrypt the server certificate. (The private key file will have been automatically generated and stored earlier if the VCS was used to produce the signing request for this server certificate.)
   - The **server private key** 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**.



Loading a trusted CA certificate onto VCS

The **Trusted CA certificate** page (**Maintenance > Certificate management > Trusted CA certificate**) allows you to manage the list of certificates for the Certificate Authorities (CAs) trusted by this VCS. Certificates presented to the VCS must be signed by a trusted CA on this list and there must be a full chain of trust to the root CA.

- To upload a new file of CA certificates, **Browse** to the required PEM file and click **Upload CA certificate**. This will replace any previously uploaded CA certificates.
- To replace the currently uploaded file with a default list of trusted CA certificates, click **Reset to default CA certificate**.



Managing certificate revocation lists (CRLs)

Certificate revocation list (CRL) files are used by the VCS to validate certificates presented by client browsers and external systems that communicate with the VCS over TLS/HTTPS. A CRL identifies those certificates that have been revoked and can no longer be used to communicate with the VCS.

We recommend that you upload CRL data for the CAs that sign TLS/HTTPS client and server certificates. When enabled, CRL checking is applied for every CA in the chain of trust.

CRL sources

The VCS can obtain CRL information from multiple sources:

- automatic downloads of CRL data from CRL distribution points
- through OCSP (Online Certificate Status Protocol) responder URIs in the certificate to be checked (SIP TLS only)
- manual upload of CRL data
- CRL data embedded within the VCS's **Trusted CA certificate** file

The following limitations and usage guidelines apply:

- when establishing SIP TLS connections, the CRL data sources are subject to the **Certificate revocation checking** settings on the **SIP** configuration page
- automatically uploaded CRL files override any manually loaded CRL files (except for when verifying SIP TLS connections, when both manually uploaded or automatically downloaded CRL data may be used)
- when validating certificates presented by external policy servers, the VCS uses manually loaded CRLs only
- when validating TLS connections with an LDAP server for remote login account authentication, the VCS uses CRL data within the **Trusted CA certificate** only

Automatic CRL updates

We recommend that the VCS is configured to perform automatic CRL updates. This ensures that the latest CRLs are available for certificate validation.

To configure the VCS to use automatic CRL updates:

1. Go to the **CRL management** page (**Maintenance > Certificate management > CRL management**).
2. Set **Automatic CRL updates** to *Enabled*.
3. Enter the set of **HTTP(S) distribution points** from where the VCS can obtain CRL files. Note that:
   - you must specify each distribution point on a new line
   - only HTTP(S) distribution points are supported; if HTTPS is used, the distribution point server itself must have a valid certificate
   - PEM and DER encoded CRL files are supported
   - the distribution point may point directly to a CRL file or to ZIP and GZIP archives containing CRL files
4. Enter the **Daily update time** (in UTC). This is the approximate time of day when the VCS will attempt to update its CRLs from the distribution points.
5. Click **Save**.

Manual CRL updates

CRL files can also be uploaded manually to the VCS. Certificates presented by external policy servers can only be validated against manually loaded CRLs.

To upload a CRL file:

1. Go to the **CRL management** page (**Maintenance > Certificate management > CRL management**).
2. Click **Browse** and select the required file from your file system. It must be in PEM encoded format.
3. Click **Upload CRL file**.
   This uploads the selected file and replaces any previously uploaded CRL file.

Click **Remove revocation list** if you want to remove the manually uploaded file from the VCS.

Note that if a certificate authority's CRL expires, all certificates issued by that CA will be treated as revoked.

Configuring revocation checking for SIP TLS connections

You must also configure how certificate revocation checking is managed for SIP TLS connections.

1. Go to the **SIP** page (**VCS configuration > SIP > Configuration**).
2. Scroll down to the **Certificate revocation checking** section and configure the settings accordingly:

Field	Description	Usage tips
Certificate revocation checking mode	Controls whether revocation checking is performed for certificates exchanged during SIP TLS connection establishment.	We recommend that revocation checking is enabled.
Use OCSP	Controls whether the Online Certificate Status Protocol (OCSP) may be used to perform certificate revocation checking.	To use OCSP, the X.509 certificate to be checked must contain an OCSP responder URI.
Use CRLs	Controls whether Certificate Revocation Lists (CRLs) are used to perform certificate revocation checking.	CRLs can be used if the certificate does not support OCSP.
Allow CRL downloads from CDPs	Controls whether the download of CRLs from the CDP URIs contained in X.509 certificates is allowed.	
Fallback behavior	Controls the revocation checking behavior if the revocation status cannot be established, for example if the revocation source cannot be contacted. *Treat as revoked*: treat the certificate as revoked (and thus do not allow the TLS connection). *Treat as not revoked*: treat the certificate as not revoked. Default: *Treat as not revoked*	*Treat as not revoked* ensures that your system continues to operate in a normal manner if the revocation source cannot be contacted, however it does potentially mean that revoked certificates will be accepted.

Appendix 1 – Troubleshooting

SIP TLS negotiation failures on neighbor and traversal zones

If **TLS verify mode** is enabled, the neighbor system's FQDN or IP address, as specified in the **Peer address** field of the zone's configuration, is used to verify against the certificate holder's name contained within the X.509 certificate presented by that system. (The name has to be contained in either the Subject Common Name or the Subject Alternative Name attributes of the certificate.) The certificate itself must also be valid and signed by a trusted certificate authority.

Therefore when certificates have been generated with peer or cluster FQDNs, ensure that the zone's **Peer address** fields are configured with FQDNs rather than IP addresses.

Appendix 2 – Certificate generation using OpenSSL only

This section describes the process for generating a private key and certificate request for the VCS 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 where the Microsoft Lync and OCS tools are not available, for test purposes, and for providing output to interact with Certificate Authorities.

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 VCS 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.

From a command prompt:

1. For Windows: change to the directory where OpenSSL is installed (typically a 'bin' directory).
   For Mac OS X: stay in the root of the user's directory.
2. For Windows: copy **openssl.cfg** to **openssl_vcs.cfg**
   For Mac OS X: copy **/system/library/openssl/openssl.cnf** to the root of the user's directory as **openssl_vcs.cfg**
3. If the certificate is for a cluster of VCSs:
   a. Use a text editor to edit the openssl_vcs.cfg file that was created by the above copy command, and ensure that the line:
      "req_extensions = v3_req # The extensions to add to a certificate request"
      has no # at the beginning of the line - delete the # if it is there.
   b. Scroll down to the "**[v3_req]**" section and below this section title add:
      subjectAltName="DNS:<FQDN of VCS cluster>,DNS:<FQDN of peer 1>,DNS:<FQDN of peer 2>,DNS:<FQDN of peer n>"

as the bottom line of this section (before) "[v3_ca]", filling in the details for the VCS deployment as appropriate (cluster FQDN and FQDNs of all peers).

c. Save the file.

No changes need to be made to the openssl_vcs.cfg file if the certificate is for a single VCS.

4. Generate a private key by running the following command:
   `openssl genrsa -out privatekey.pem 2048`
   The **privatekey.pem** file will be used to create the certificate request and will also be required for loading into the VCS. The file is created in the directory that the openssl command is run from.
5. Generate a certificate request (suitable for use with Microsoft Certification Authority) by running the following command:
   `openssl req -new -key privatekey.pem -config openssl_vcs.cfg -out certcsr.der -outform DER -sha1`
6. Enter the data requested, including:
   - Country
   - State or province
   - Locality name
   - Organization name
   - Organizational unit
   - Common name - this is the VCS cluster FQDN if the certificate is for a cluster of VCSs or it is the FQDN of the VCS if the certificate is for a single VCS
   - Email address - optional, can leave blank
   - A challenge password - optional, can leave blank
   - An optional company name - optional, can leave blank

   After entering the requested data, the certificate request file **certcsr.der** is now available.

To validate that DNS entries have been entered correctly into the request, the **certcsr.der** file can be decoded using the command:

`openssl req -text -noout -in certcsr.der -inform DER`

This certificate request file can be passed to an internal or third-party Certificate Authority for generating the X.509 certificate. OpenSSL can be used to operate a private CA, as described below.

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.

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 three files/directories required to be created are:

- A **demoCA** directory in the current directory.
- An empty file called **index.txt** in the **demoCA** directory.
- A file called **serial** in the **demoCA** directory, storing the current serial number of an issued certificate. This should contain a 4-digit hexadecimal number, such as "1000".

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. For Windows: change to the directory where OpenSSL is installed (typically a 'bin' directory).
   For Mac OS X: stay in the root of the user's directory.
2. Generate a private key for the CA by running the following command:
   `openssl genrsa -aes256 -out ca.key 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 ca.key file will be used to create the CA certificate and to sign other certificates and must also be kept secure.
3. Generate the CA certificate by running the following command.
   For Windows: **`openssl req -new -x509 -key ca.key -config openssl.cfg -sha1 -extensions v3_ca -out ca.crt`**
   For OS X: **`openssl req -new -x509 -key ca.key -sha1 -extensions v3_ca -out ca.crt`**
4. Enter 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 **ca.crt** 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. Convert the previously generated certificate request to PEM format by running the following command:
   `openssl x509 -in certcsr.der -inform DER -out certcsr.pem -outform PEM`
2. Generate a signed server certificate by running the following command:
   `openssl ca -outdir . -config openssl.cfg -cert ca.crt -keyfile ca.key -in certcsr.pem -out server.pem -md sha1`
3. You will be prompted to enter the password for the CA's private key.

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

Creating self-signed certificates using OpenSSL

A self-signed certificate does not use a CA, and therefore there is no chain of trust behind it. However, a self-signed certificate can be a quick way of deploying a certificate for encryption and, if it can be exchanged securely to the client, can be used for individual identity verification. Self-signed certificates are not appropriate for production deployments but may be useful in a test deployment. Instructions on creating self-signed certificates are provided below.

A CA certificate is essentially a self-signed certificate, since it is the root of the chain of trust (there is no other authority to verify it). For a self-signed certificate, however, no passkey is required when creating the key, since the server requires access to the key for communications.

From a command prompt:

1. For Windows: change to the directory where OpenSSL is installed (typically a 'bin' directory).
   For Mac OS X: stay in the root of the user's directory.
2. Generate a private key by running the following command:
   `openssl genrsa -out privatekey.pem 2048`
   The **privatekey.pem** encrypts communications and will need to be copied onto the VCS. The file is created in the directory that the openssl command is run from.
3. Generate the CA certificate by running the following command:
   For Windows: **`openssl req -new -x509 -key privatekey.pem -config openssl.cfg -out server.pem`**
   For OS X: **`openssl req -new -x509 -key privatekey.pem -out server.pem`**
4. Enter 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

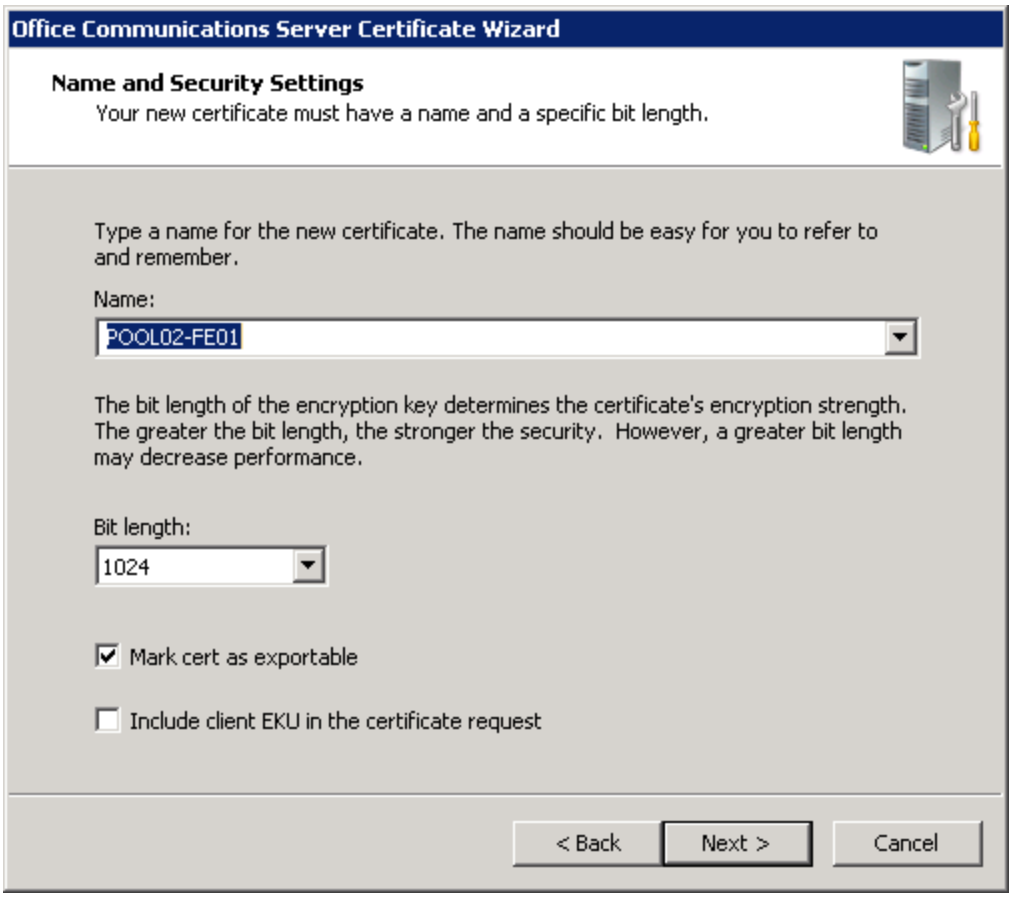
After entering the requested data, the operation completes and the server certificate **server.pem** is now available. The private key is stored in **privatekey.pem**.

Appendix 3 – Certificate generation using Microsoft OCS

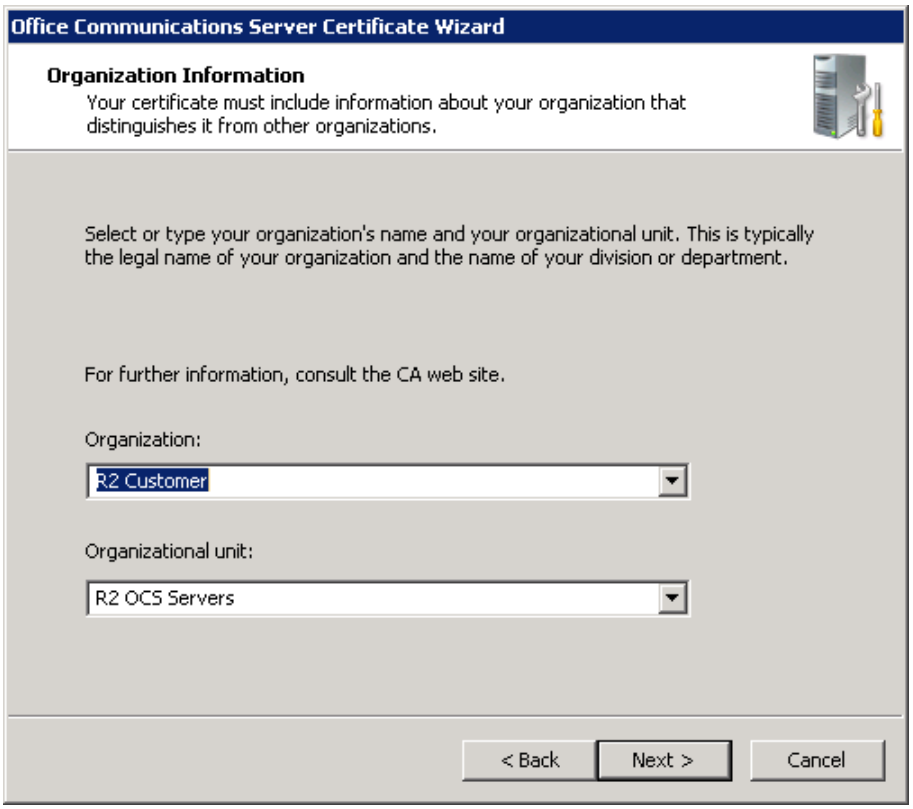
Generating a certificate request

To obtain a private key, root (CA) certificate and the server / client certificate using OCS:

1. On OCS, select **Start > Administrative Tools > Office Communications Server 2007**.
2. Expand **Enterprise pools**, if available, otherwise expand **Standard Edition Servers**.
3. Expand **OCS Pool**.
4. Select the specific OCS.
5. Click **Certificates** and follow the wizard, as described in the following steps.
6. On the **Name and Security Settings** page:
   a. For the **Name**, select the relevant pool (usually default value).
   b. Select a **Bit length** of 2048.
   c. Select **Mark cert as exportable**.
   d. Do not include EKU in the certificate request.



7. Leave **Organization Information** entries as default.



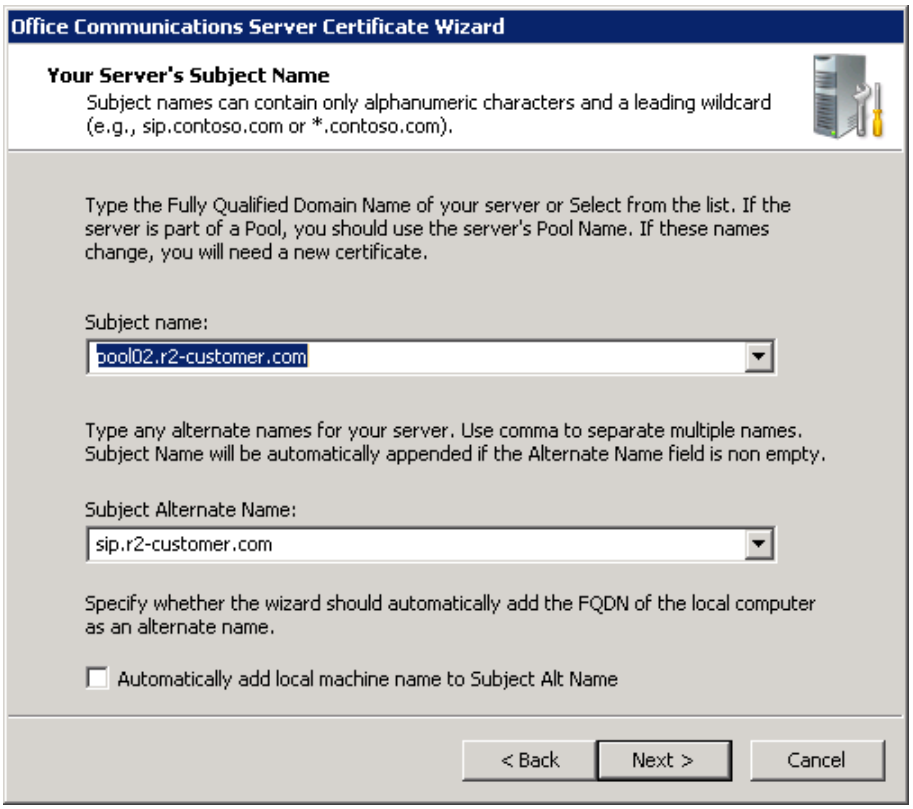
8. Specify **Subject Name** and **Subject Alternate Name** details.
   If the certificate is for a VCS in a cluster, either:
   - Create a certificate that can be loaded onto every peer of the cluster.
   - Create individual certificates for each VCS peer.

   If creating a certificate that can be loaded onto every peer of the cluster:
   - Subject name = VCS cluster's FQDN, e.g.vcscluster1.test-customer.com
   - Subject Alternate Name = a comma separated list of the VCS cluster's FQDN and the VCS peers' FQDNs (DNS Local hostname concatenated with DNS Domain) e.g. vcscluster1.test-customer.com, vcspeer1.test-customer.com, vcspeer2.test-customer.com

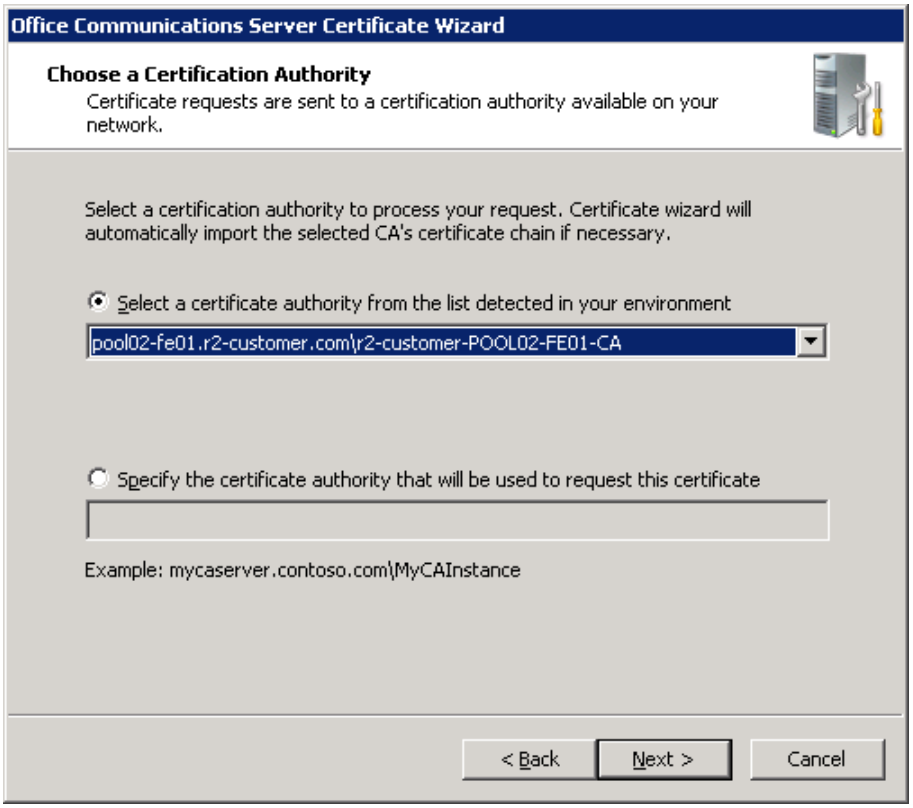
   If creating individual certificates for each VCS peer:
   - Subject name = VCS peer's FQDN (DNS Local hostname concatenated with DNS Domain) e.g. vcspeer1.test-customer.com
   - Subject Alternate Name = a comma separated list of the VCS cluster's FQDN and the VCS peer's FQDN, e.g. vcscluster1.test-customer.com,vcspeer1.test-customer.com

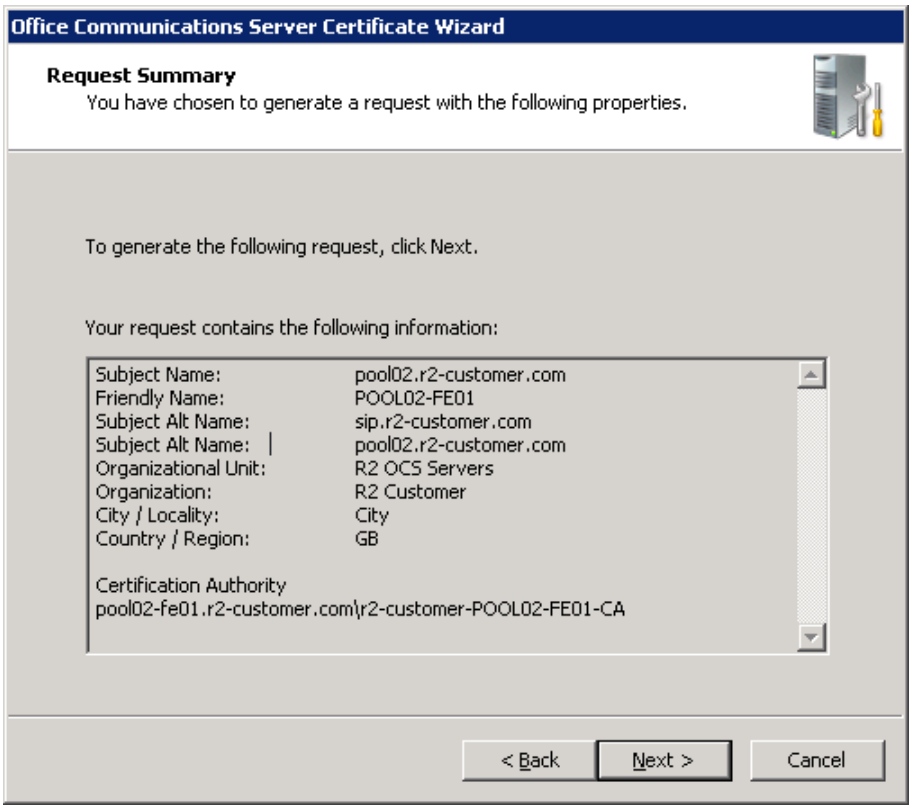
   If the certificate is for a standalone VCS:
   - Subject name = VCS's FQDN (DNS Local hostname concatenated with DNS Domain) e.g. vcs.test-customer.com
   - Subject Alternate Name = blank

   Do not select **Automatically add local machine name to Subject Alt Name** (this is for the VCS, not the OCS PC).
   When prompted that "Subject name does not match pool name … continue?". Click **Yes**.



9. Add geographical information when prompted.





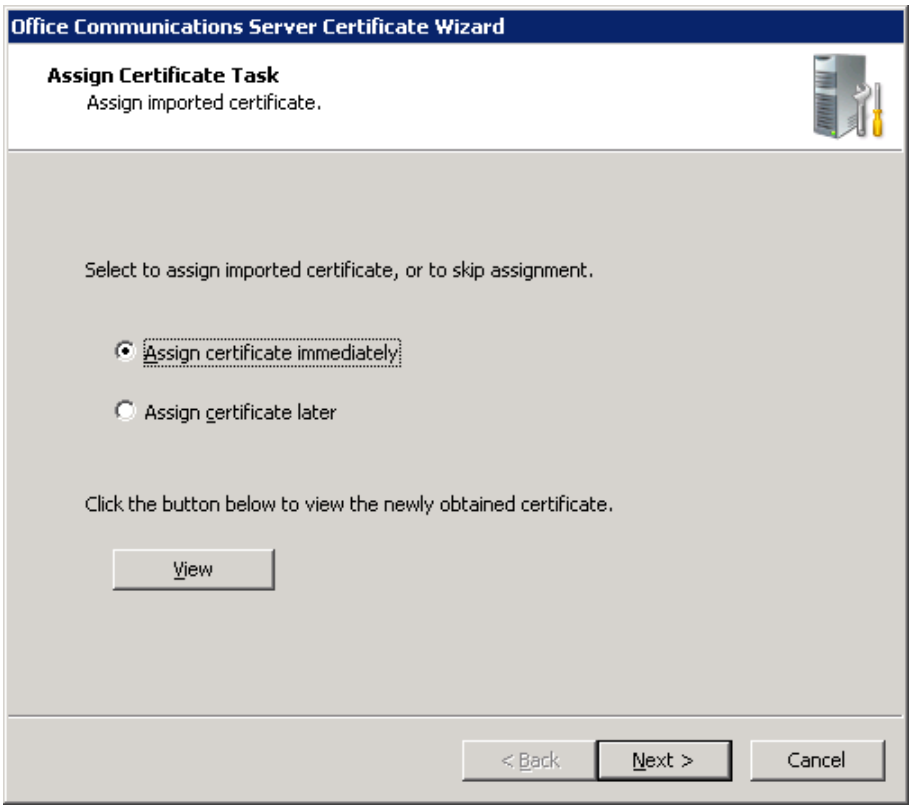
10. Stop at **Assign certificate immediately**.

The certificate request has now been generated. The next stage is to authorize the request.

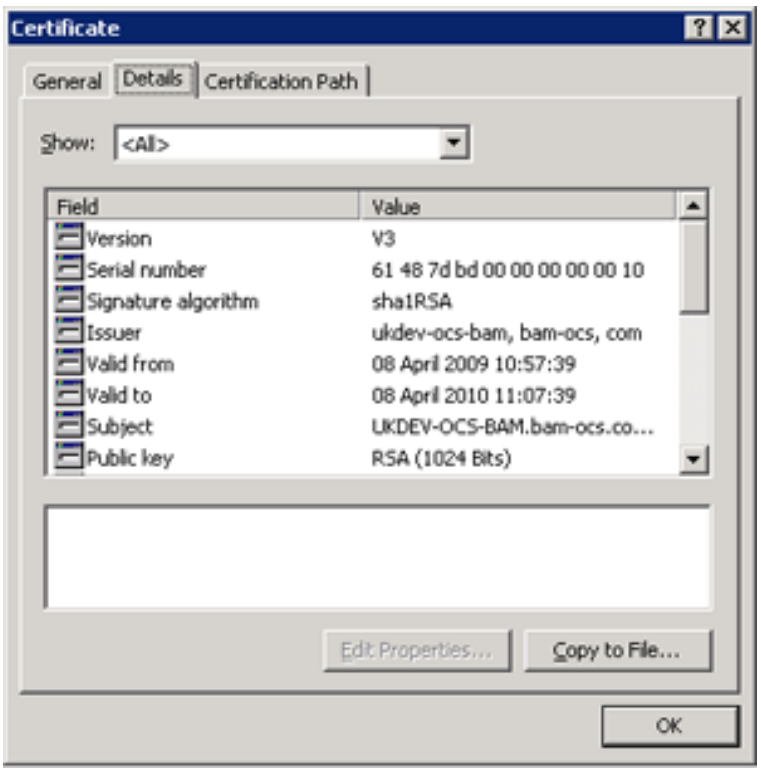
Authorizing a certificate request and generating a PEM certificate file using OCS

The certificate can be created on OCS:

1. Select **Start > All Programs > Administrative Tools > Certification Authority**.
2. Choose the relevant Local Certification Authority and select **Pending Requests**.
3. Right-click on the certificate request generated above, select **All tasks**.
4. Follow the wizard until you get to the **Assign imported certificate** step.
5. Click **View**.



6. Select the **Details** tab, then click **Copy to File**.

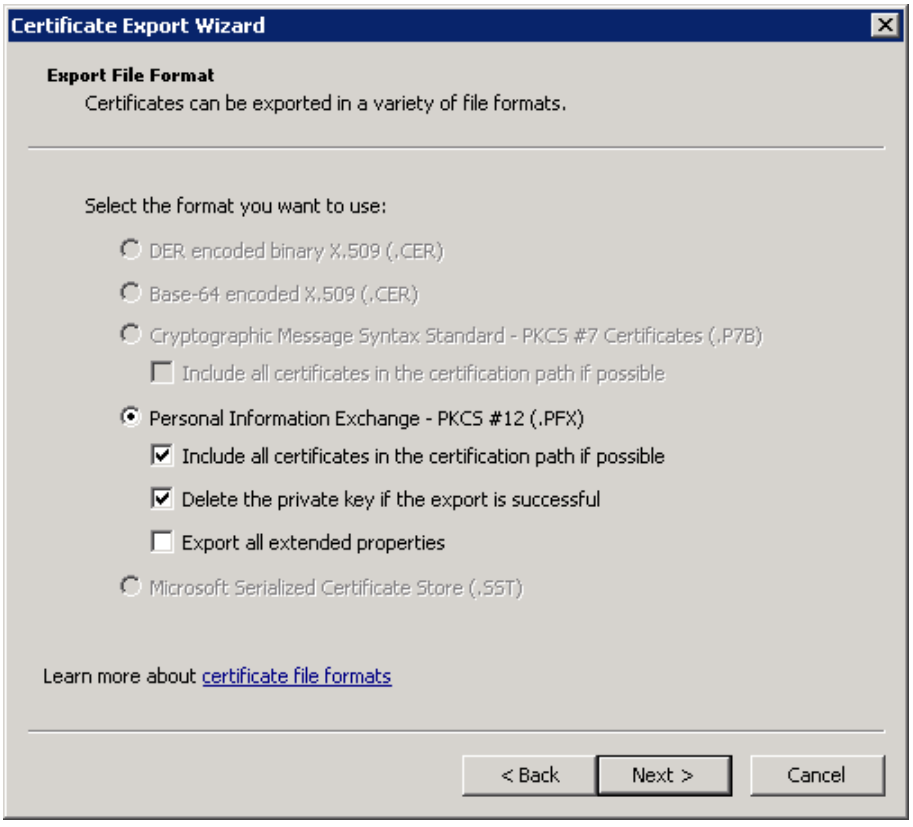


The Certificate Export Wizard opens.

7. Click **Next**.

8. Ensure that **Yes, export the private key** is selected and click **Next**.

9. Ensure that the selections are as shown below (note that the order of the tick boxes may differ in different versions of Windows), then click **Next**.



10. Enter a **Password** that will be used to encrypt the private key (remember this as it will be needed to decrypt the private key certificate later) and click **Next**.

11. Enter the path and **File name** to save the certificates file (the file should have the extension .pfx) and click **Next**.
12. Click **Finish**.

13. Certificate export was successful – click **OK**.
    You are returned to the **Certificate Details** page.
14. Click **OK** to close.
    You are returned to the Assign Certificate wizard:
15. Click **Cancel**.
    Do NOT assign the certificate.

The certificate has now been generated and exported as a .pfx file, for example **cert_inf.pfx** (the .pfx contains the private key, the server certificate and root (CA) certificate in pkcs12 format).

On a Linux (Windows or other) system running openssl:

1. Copy the .pfx file to the system where openssl can be run.
2. Execute the command line:
   `openssl pkcs12 -in cert_inf.pfx -out cert_inf.pem`
3. Type in the password that was used during the cert_inf.pfx generation.
4. Enter and verify a PEM pass phrase (to be used in the next stage – removing the password from the private key).
5. Continue with Process the 'cert_inf.pem' file into server certificate, CA certificate and private key [p.23] below.

Process the 'cert_inf.pem' file into server certificate, CA certificate and private key

The '.pem' file contains 3 sections:

1. Private key section:

```
Bag Attributes
…
Key Attributes
    X509v3 Key Usage: 10
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,CAD24BC3A702A939
l5JTMK+Irjx2ptUWXbqEgaGwOE6UiXJE+x4Ga7sc9MXB2fvzHNehiyFHS+FoqFZK
…
7vGI/4Ezt5Stajm/p+ENGD+jMstT3a3Q85SnfvwBGVtdleKFUZ0E4Q==
-----END RSA PRIVATE KEY-----
```

2. Server certificate:

```
Bag Attributes
    localKeyID: 01 00 00 00
    friendlyName: vcs-certificate
subject=/C=NO/L=Oslo/O=net2/OU=ast/CN=vcs.net2.int
issuer=/DC=int/DC=net2/CN=net2-CA
-----BEGIN CERTIFICATE-----
MIIE3DCCA8SgAwIBAgIKI/Co1QAAAAAADzANBgkqhkiG9w0BAQUFADA9MRMwEQYK
…
6aRy7KXfeBLN/pqBEVSjjlCljmXa5hc5AGmzyTfrSRXviHE3qpmT0Lnld31f6qGK
-----END CERTIFICATE-----
```

   Note that the issuer specifies the issuer that approves this Server certificate, and the server certificate typically has a 'friendlyName' in the header.

3. CA certificate:

```
Bag Attributes: <Empty Attributes>
subject=/DC=int/DC=net2/CN=net2-CA
issuer=/DC=int/DC=net2/CN=net2-CA
-----BEGIN CERTIFICATE-----
MIIDVTCCAj2gAwIBAgIQQwRnazGMG5JGOacRYvwNlTANBgkqhkiG9w0BAQUFADA9
…
tQy4Hfj50yemURZ7mFCXGapcegsOC5/WDhOcIrlkcDJ2lcvBgUj4rbI=
-----END CERTIFICATE-----
```

The root CA certificate is a self signed certificate; the subject and issuer of the certificate are the same. Where there is an external certificate authority, the root CA certificate can be used from the external certificate authority.

Certificates can have hierarchical trust:

- The server certificate is trusted by (created using) a secondary certificate.
- The secondary certificate is trusted by (created using) the root CA certificate.
- The root CA certificate is self signed.

In this case, 4 entries will be found in the created PEM file.

The CA file loaded onto the VCS should consist of the full CA chain between the root CA and the server certificate (concatenated in a single file) – because to keep the hierarchy and be able to authenticate the server certificate, both of these certificates are needed. (See Example certificate with a root CA and secondary CA [p.29] .)

1. Using a text editor, edit the 3 sections of the PEM file into separate files:
   - The private key section: to a file, for example, **priv_e.pem**
   - Server certificate section: to a file, for example, **server.pem**
   - CA certificate section: to a file, for example, **root_ca.pem**
2. Decrypt the private key using the command line:
   `openssl rsa -in priv_e.pem -out priv.pem`
3. Enter the PEM pass phrase.
4. Copy the three files to the PC which is going to be used to configure the VCS.

Appendix 4 – 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 VCS. 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

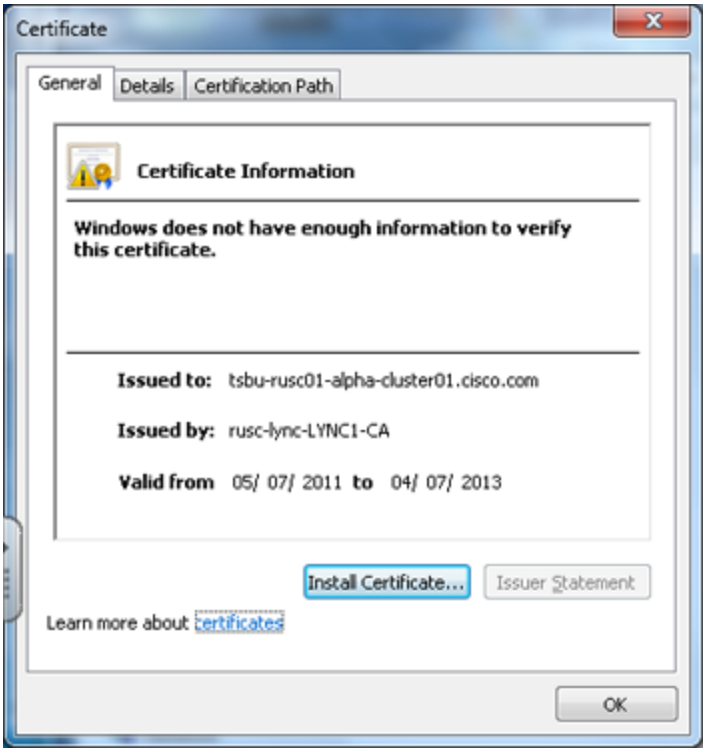
To convert from DER to PEM format, on a system running openssl, execute the command:

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

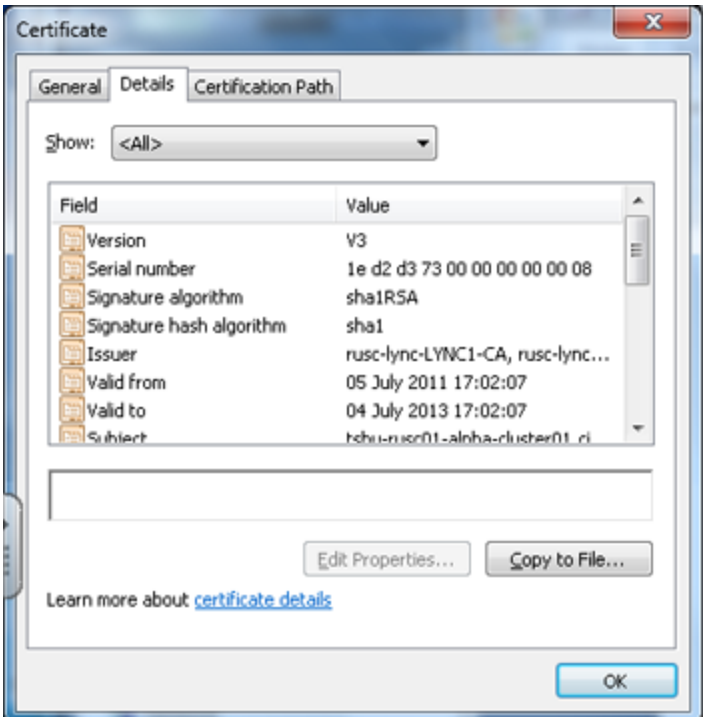
Converting a DER certificate file to a PEM file using Microsoft Windows

To convert from DER to PEM format using Microsoft Windows:

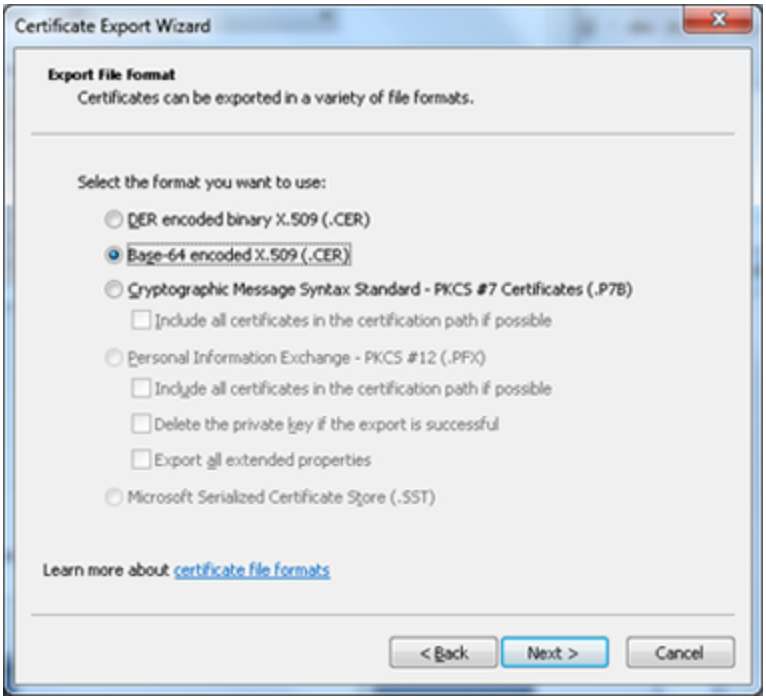
1. Double click on the DER file to convert (this will likely have a '.cer' extension).



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 VCS [p.8] section of this document.

Appendix 5 – Example certificates

Example certificate with self-signed root CA

This certificate contains:

- private key
- server certificate
- root CA (self signed)

```
Bag Attributes
    Microsoft Local Key set: <No Values>
    localKeyID: 01 00 00 00
    friendlyName: le-d2c14df5-fb1f-459e-a269-dee27900c015
    Microsoft CSP Name: Microsoft RSA SChannel Cryptographic Provider
Key Attributes
    X509v3 Key Usage: 10
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,CAD24BC3A702A939

l5JTMK+Irjx2ptUWXbqEgaGwOE6UiXJE+x4Ga7sc9MXB2fvzHNehiyFHS+FoqFZK
6XrvdcB4vUrUC/PEhiBfdezVYqiYfEZRdtiFTKL9xJZ6TNo5qOcXo9kXE2gut/uc
…
8sIBOWKErI1LQBJFhaZl117KiC1j4vy+9sFyUtm+CVw72CU4HBYIv2wJM169m0FK
7vGI/4Ezt5Stajm/p+ENGD+jMstT3a3Q85SnfvwBGVtdleKFUZ0E4Q==
-----END RSA PRIVATE KEY-----
Bag Attributes
    localKeyID: 01 00 00 00
    friendlyName: vcs-certificate
subject=/C=NO/L=Oslo/O=net2/OU=ast/CN=vcs.net2.int
issuer=/DC=int/DC=net2/CN=net2-CA
-----BEGIN CERTIFICATE-----
MIIE3DCCA8SgAwIBAgIKI/Co1QAAAAAADzANBgkqhkiG9w0BAQUFADA9MRMwEQYK
CZImiZPyLGQBGRYDaW50MRQwEgYKCZImiZPyLGQBGRYEbmV0MjEQMA4GA1UEAxMH
…
wtkRwhL8xcuMHQybEDRcPZlhxDNGS9oORSCVnnpqZkbbh6fmRJNqYGjk6hM9dgMW
6aRy7KXfeBLN/pqBEVSjjlCljmXa5hc5AGmzyTfrSRXviHE3qpmT0Lnld31f6qGK
-----END CERTIFICATE-----
Bag Attributes: <Empty Attributes>
subject=/DC=int/DC=net2/CN=net2-CA
issuer=/DC=int/DC=net2/CN=net2-CA
-----BEGIN CERTIFICATE-----
MIIDVTCCAj2gAwIBAgIQQwRnazGMG5JGOacRYvwNlTANBgkqhkiG9w0BAQUFADA9
MRMwEQYKCZImiZPyLGQBGRYDaW50MRQwEgYKCZImiZPyLGQBGRYEbmV0MjEQMA4…
EGGKPlsJjToyU2E9s2yZHU1XJ7QEGDqUETmeqL3I5Bk9hXE2SotJQ1OQaYwA07MY
tQy4Hfj50yemURZ7mFCXGapcegsOC5/WDhOcIrlkcDJ2lcvBgUj4rbI=
-----END CERTIFICATE-----
```

Example certificate with a root CA and secondary CA

This certificate contains:

- private key
- server certificate
- root CA (self signed)
- secondary root certificate (signed by root CA)

```
Bag Attributes
    1.3.6.1.4.1.311.17.2: <No Values>
    localKeyID: 01 00 00 00
    friendlyName: le-7430b4bb-13af-45c5-832d-067edcee82f3
    Microsoft CSP Name: Microsoft RSA SChannel Cryptographic Provider
Key Attributes
    X509v3 Key Usage: 10
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,6C4F771351A95DA5

eqFef6VcBEUeD3LxeyHgFyW2nOqXnbTyEldT5FswIxZAadFRfFFqE2xsozVSaaeX
/SQ3FF6lcZeW6uKTYm4ImyFKc/RNFGZv0dXEnVsVia0HWbkDeDJ8ZbyFRqKEEJaJ
…
bHIpbVYITKHmASDs0RyybY9U8y4WE6N+6F2PJWKEIWo4oeKO0vyCfQAS+q3bb8LL
Sj5XaxFzd15XDmNrkwlUINumM5LxHhAUvmo7oiiqdRTyB+57p0+rjg==
-----END RSA PRIVATE KEY-----
Bag Attributes
    localKeyID: 01 00 00 00
    1.3.6.1.4.1.311.17.3.20: B0 D2 22 E9 68 FC 0F A3 CE 17 54 95 DB 9C 65 F9 67 FF C4 69
    1.3.6.1.4.1.311.17.3.71: 4A 00 54 00 53 00 58 00 57 00 30 00 32 00 32 00 2E 00 61 00
75 00 73 00 74 00 72 00 69 00 61 00 2E 00 6C 00 6F 00 63 00 61 00 6C 00 00 00
    friendlyName: vcsctelepresence.myco.local
subject=/C=AT/ST=Vienna/L=Vienna/O=Myco TA AG/OU=ICT/CN=vcsctelepresence.myco.local
issuer=/serialNumber=35126/CN=Myco TA local 01/emailAddress=pki@email.com/O=Myco TA
AG/C=AT
-----BEGIN CERTIFICATE-----
MIIGAjCCBOqgAwIBAgIKJFzWrwAAAAC6hjANBgkqhkiG9w0BAQUFADCBgjEOMAwG
A1UEBRMFMzUxMjYxJDAiBgNVBAMTG1RlbGVrb20gQXVzdHJpYSBUQSBsb2NhbCAw
…
UAzGHbbIvX1T6kVLRTJ7heXz0BhX5ar3pWP6D7lUhGkQB+hfluUdzuePMoQaNIAG
H/LV1efZ
-----END CERTIFICATE-----
Bag Attributes
    1.3.6.1.4.1.311.17.3.20: 24 7D 5D 14 7E 6D 8D B0 88 24 9E DD 77 CF 77 C7 20 06 06 94
    1.3.6.1.4.1.311.17.3.75: 36 00 42 00 43 00 43 00 45 00 43 00 38 00 43 00 39 00 31 00
30 00 44 00 41 00 46 00 35 00 31 00 31 00 31 00 34 00 43 00 44 00 43 00 38 00 36 00 44 00
37 00 31 00 41 00 37 00 34 00 31 00 36 00 5F 00 00 00
subject=/serialNumber=32618/CN=eSignature TA Basic/OU=eSignature TA Basic/O=Myco TA
AG/C=AT
issuer=/serialNumber=32618/CN=eSignature TA Basic/OU=eSignature TA Basic/O=Myco TA
AG/C=AT
-----BEGIN CERTIFICATE-----
MIIFtTCCBJ2gAwIBAgICf2owDQYJKoZIhvcNAQEFBQAweTEOMAwGA1UEBRMFMzI2
MTgxHDAaBgNVBAMTE2VTaWduYXR1cmUgVEEgQmFzaWMxHDAaBgNVBAsTE2VTaWdu
```

```
…
b1RT/9LLr+7ly7kC2EmCYTEXTtgakin1Rx+cLA7YtT62WVFSGSISP06INQ7HXEuc
dxxKC8ccBZSlsV6eRUE5ZVSn3jIv1ALzRA==
-----END CERTIFICATE-----
Bag Attributes
    1.3.6.1.4.1.311.17.3.20: 3B AE F1 8D F2 9D 61 53 40 78 F9 81 00 F7 A7 B4 CB 27 53 1B
    1.3.6.1.4.1.311.17.3.75: 36 00 42 00 43 00 43 00 45 00 43 00 38 00 43 00 39 00 31 00
30 00 44 00 41 00 46 00 35 00 31 00 31 00 31 00 34 00 43 00 44 00 43 00 38 00 36 00 44 00
37 00 31 00 41 00 37 00 34 00 31 00 36 00 5F 00 00 00
subject=/serialNumber=35126/CN=Myco TA local 01/emailAddress=pki@telekom.at/O=Myco TA
AG/C=AT
issuer=/serialNumber=32618/CN=eSignature TA Basic/OU=eSignature TA Basic/O=Myco TA
AG/C=AT
-----BEGIN CERTIFICATE-----
MIIGSDCCBTCgAwIBAgIDAIk2MA0GCSqGSIb3DQEBBQUAMHkxDjAMBgNVBAUTBTMy
NjE4MRwwGgYDVQQDExNlU2lnbmF0dXJlIFRBIEJhc2ljMRwwGgYDVQQLExNlU2ln
…
xYSc88bZVf5JooXMImP5kPFVgLsPW5BuWgC6VIdTnopSJrYYCD65oxy/rwL+00G9
XK9EL6BuFqa7+4MZlyEVxJspqCZOfThr86IEHg==
-----END CERTIFICATE-----
```

Appendix 6 – 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

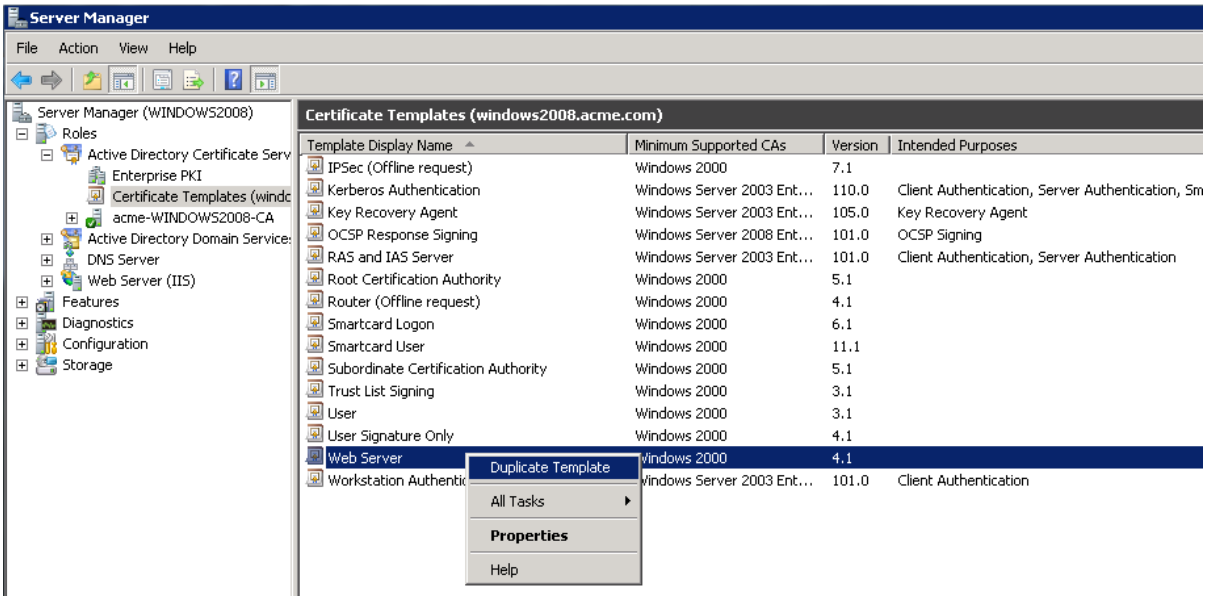
The certificate in use for a website being visited can be viewed in Internet Explorer by clicking the lock icon to the right of the address bar. A **Website Identification** dialog will appear. Click the **View Certificates** link at the bottom.

Appendix 7 – Configuring Windows Server Manager with a "client and server" certificate template

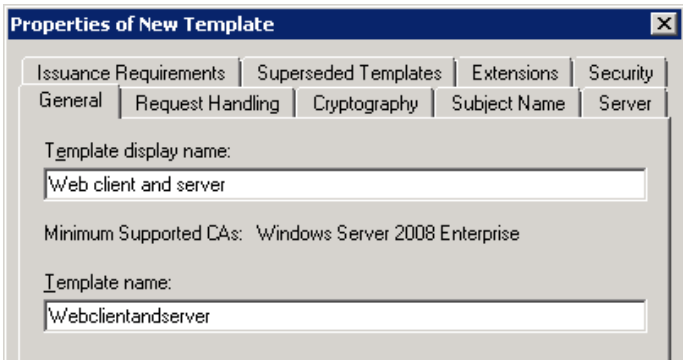
The default "Web Server" certificate template used by the Microsoft Certification Authority application will only create a certificate for Server Authentication. The server certificate for the VCS also needs Client Authentication if you want to configure a neighbor or traversal zone with mutual authentication (where **TLS verify mode** is enabled).

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

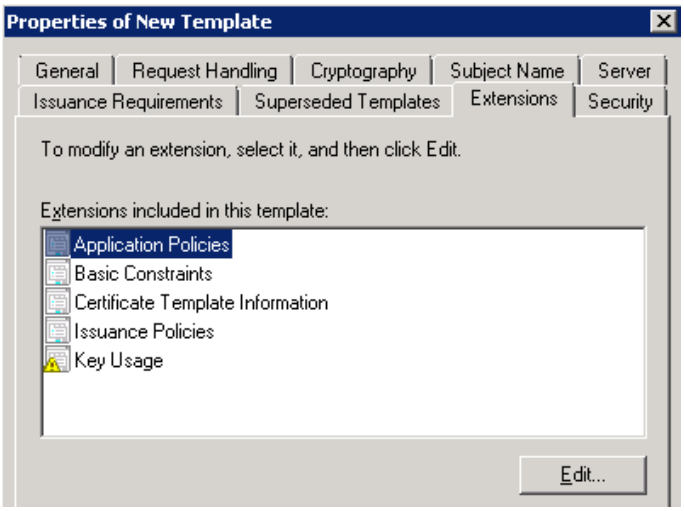
1. In Windows, launch **Server Manager** (**Start > Administrative Tools > Server Manager**).
2. Expand the **Server Manager** navigation tree to **Roles > Active Directory Certificate Services > Certificate Templates (<domain>)**.
3. Right-click on **Web Server** and select **Duplicate Template**.



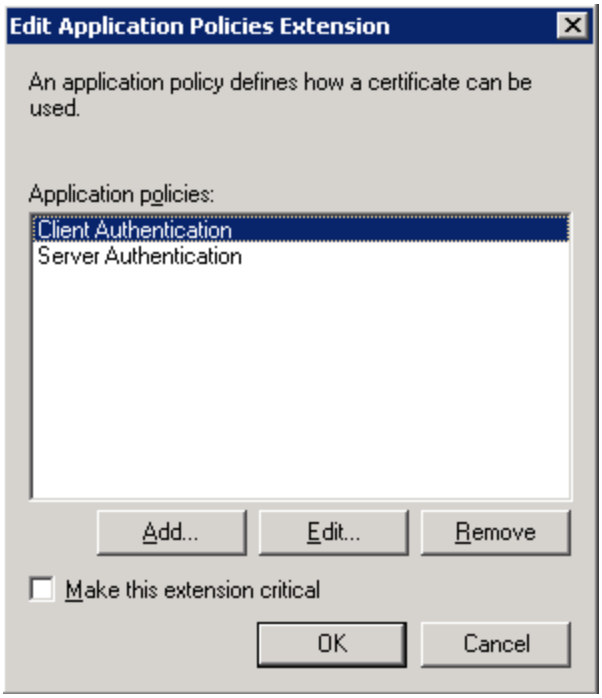
4. Select the appropriate **Windows Server Enterprise** version and click **OK**.
5. On the **General** tab, enter the **Template display name** and **Template name**, for example **`Web client and server`** and **`Webclientandserver`**.



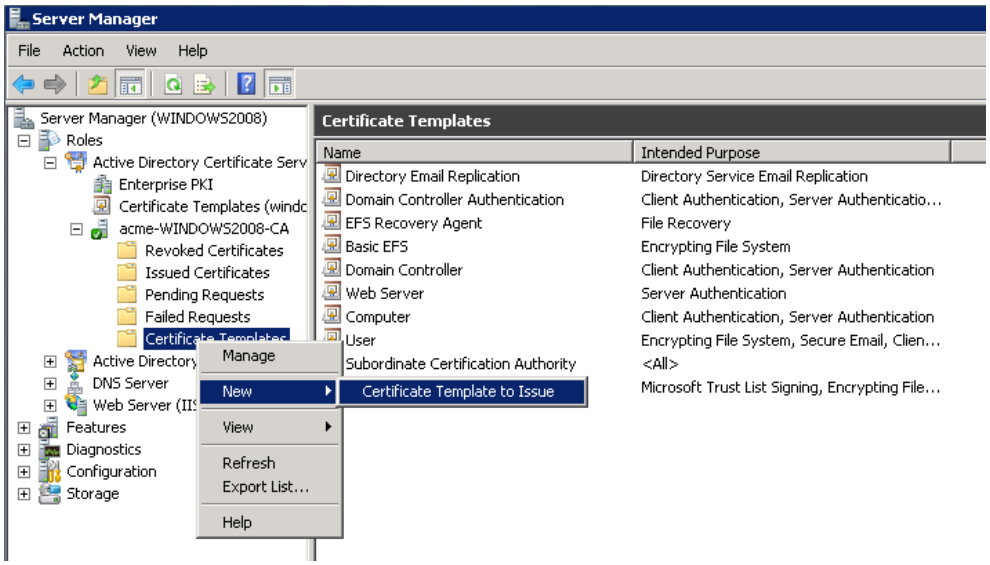
6. On the **Extensions** tab, select **Application Policies** and click **Edit**.



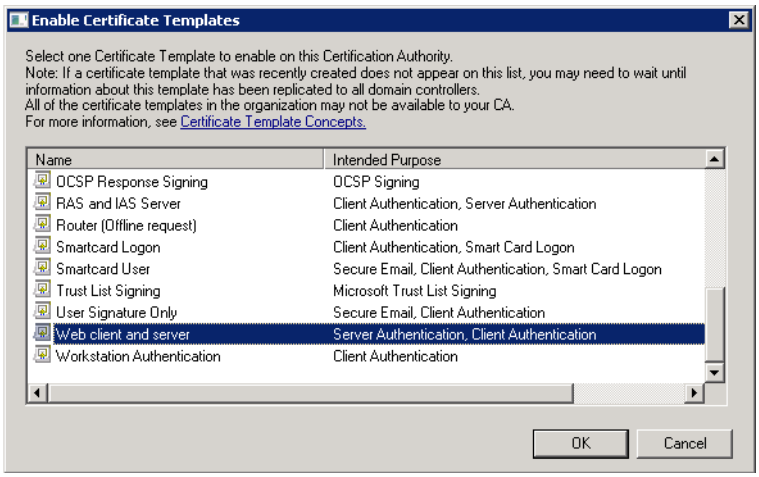
7. Add **Client Authentication** to the set of application policies:
   a. Click **Add**.
   b. Select **Client Authentication** and click **OK**.
   c. Click **OK**.



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
1	November 2009	Initial release.
2	October 2010	New document styles applied. New appendices added for decoding certificates and guidance on generating certificates for use with Microsoft Office Communications Server (OCS).
3	September 2011	Updated for Microsoft Lync 2010 (Lync).
4	December 2011	Minor updates for clarification.
5	February 2012	Major clarifications and updates, including OpenSSL-specific section.
6	August 2012	Updated for VCS X7.2 functionality to generate certificate signing requests.
7	February 2013	Added sections on CRL management, troubleshooting, and how to configure Windows Server Manager with a "client and server" certificate template.